UNIVERSIDAD CARLOS III DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

DEPARTAMENTO DE INGENIERIA ELÉCTRICA

Diseño de una interfaz para la medida del desfase de dos señales senoidales

INGENIERIA TÉCNICA INDUSTRIAL:
ELECTRÓNICA INDUSTRIAL

AUTOR: ANTONIO JOSÉ CARRASCO DÍAZ
TUTOR: SIMÓN RAFAEL DÁVILA SOLANO

a 10 de JUNIO 2009

Agradecimientos

Esta memoria va dedicada a todas las personas que siempre me han apoyado a lo largo de toda la carrera y el proyecto.

Mención especial merece Daniel Estrella que siempre me ha apoyado y ayudado. También quiero agradecerle a Aranzazu su paciencia y su comprensión. A mis amigos más cercanos como Agustín Martín, Agustín Arenas. Mi familia por ayudarme con la logística y su apoyo.

En fin a todos los que habéis estado cerca y no tan cerca, pero siempre arrimando el codo.

Índice

Índice de Figuras

Índice de Tablas

1. Introducción

En un mundo en el que la energía tiene un valor tan alto, cualquier pérdida de ésta puede suponer un desembolso considerable de dinero, y lo que es peor aún, se puede llegar a una situación en la que un deterioro del canal o en los centros de transformación, puede llegar a provocar un incendio, apagón, etc., si no se tiene un mantenimiento preciso de los elementos que componen el sistema.

Por ello, el desarrollo de un equipo que pueda ser capaz de ver el desgaste de los transformadores con el tiempo, para que puedan ser reemplazados por otros equipos a tiempo de que no se produzca pérdida de energía o males mayores, es una necesidad.

En el presente proyecto se va a realizar un medidor de desfase entre dos señales senoidales cuya aplicación puede ser múltiple. Sin embargo, se podría adaptar para que formase parte de un equipo de las características que se han descrito anteriormente.

El equipo que comprueba el estado de funcionamiento del transformador se divide en dos partes.

- ***Fuente de alimentación***. Esta **primera parte**, pretende generar una señal de voltaje lo suficientemente alta como para alimentar el transformador y con un frecuencia de oscilación, lo suficientemente baja como para poder analizar las pérdidas debidas al aislante.
- ***Medidor de desfase***. Esta **segunda parte** del equipo que es la que se va a desarrollar en el presente proyecto fin de carrera, trata desarrollar una interfaz que calcule el desfase de dos señales senoidales, y que en el equipo de aplicación práctica, el desfase es el cálculo de la tg δ que representaría el desgaste del dieléctrico de un condensador. Además para obtener dichas señales, se va a diseñar un sistema de adquisición de datos de esas dos señales senoidales.

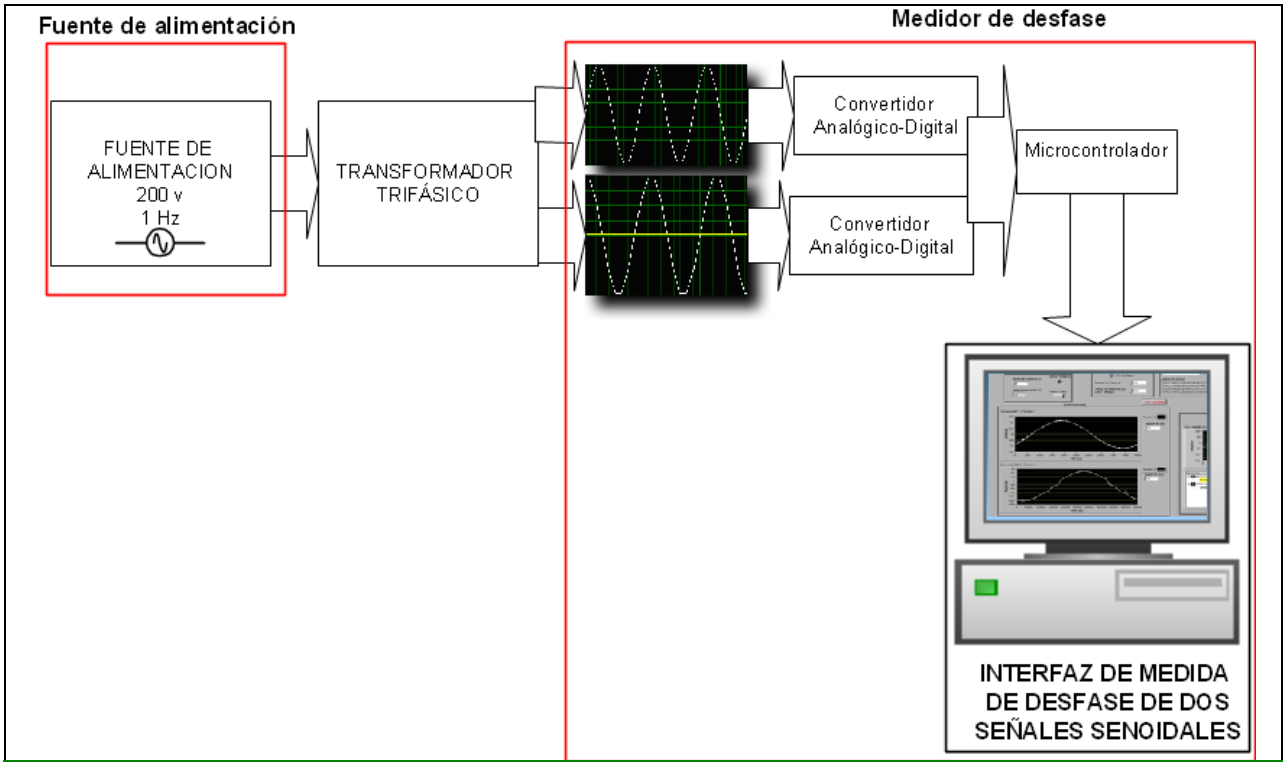


Fig. 1: Esquema general del equipo de medición de desgaste de transformador

Metodología

El sistema de adquisición de señal y posterior procesamiento de la señal que se presenta, cuenta con la ventaja de ser un sistema de bajo coste adaptado a cualquier tipo de entorno.

Los sistemas de adquisición de datos actuales, utilizan grandes y caras tarjetas de datos que permiten hacer medidas muy exactas y de muchas señales a la vez.

Lo que se ha intentado con este proyecto ha sido realizar un pequeño sistema de procesamiento de datos que tiene como aplicación calcular el desfase entre dos señales senoidales.

Fig. 2: Fases del desarrollo del sistema de medición

Para llevar a cabo el medido de desfase se necesita dividir el proyecto en sus partes fundamentales que son: la **adquisición de datos**, el **procesado** y **envío de datos al PC**, y desde éste **calcular el desfase** y registrar los datos obtenidos.

El método seleccionado para el cálculo del desfase es un método clásico, que consiste en muestrear las dos señales y calcular el intervalo de tiempo de paso por un punto de referencia de cada una de las dos señales. Una vez obtenido el tiempo, calculamos el desfase.

La primera parte se va a llevar a cabo mediante dos conversores ADC que adquieren una señal analógica en un rango de 0-5 V y en 100 μS convierten dicha señal en digital que envían al microcontrolador PIC.

El microcontrolador PIC, recoge los datos según la configuración establecida desde el PC, de tal forma que las señales de orden habilitan y deshabilitan los ADC, según la frecuencia y número de muestras que se soliciten.

Se decidió utilizar una interfaz de conexión mediante USB. Esto además puede proporcionar suficiente energía como para alimentar el sistema de medición.

Una vez seleccionado es el microcontrolador, se necesita seleccionar el programa codificador y el lenguaje utilizado, que en ambos casos es el lenguaje ‘C’.

Limitaciones

El sistema de adquisición de datos que se ha construido, nos permite tener un sistema autónomo de medida con el que recoger los datos, configurando simplemente la cantidad de muestras y el tiempo que se necesite, para poder obtener los datos de desfase y estados de las señales de entrada al sistema de medida.

Descripción	Valor	Unidades
Tensión de Alimentación (Conexión USB)	5	V
Corriente consumida	250	mA (máx)

Tabla 1: Características energéticas del medidor

Descripción	Valor	Unidades
Rango de medida	1- 2500	Hz
Error de desfase aproximado	± 5	grados
Tiempo entre muestras mínimo	330	µs
Señal de entrada analógica	0-5 Vrms	V
Tiempo de captura de datos (Interfaz)	2 - 100	Segundos

Tabla 2: Características del medidor

En esta tabla se muestras las características principales del sistema de medida. Como se puede observar, la alimentación del dispositivo es mantenida a través de la conexión USB. Pues tanto los dispositivos de adquisición como el microcontrolador demandan poca cantidad de corriente y es suficiente con los 250mA que es capaz de dar como máximo el puerto USB de un ordenador.

Además de todo ello se tiene la posibilidad de registrar los datos en un archivo de texto plano para su posterior procesado con otros programas. En dicho archivo, se muestra toda la información necesaria relativa a la medida como: grado de desfase, frecuencia de muestreo, número de muestras utilizada, intervalo de recogida, fecha y hora de creación del archivo.

1.1. Descripción de capítulos

La memoria se va a organizar con los siguientes contenidos:

Capítulo 1: este capítulo muestra una introducción sobre el objetivo del medidor y de qué consta.

Capítulo 2: este capítulo se describe brevemente un marco teórico necesario para el desarrollo de proyecto.

Capítulo 3: es la parte principal del libro pues se trata de una descripción general de todo el sistema de medida del desfase. Se divide a su vez en tres bloques:

Capítulo 3.1: Bloque I: contiene una descripción del sistema de conversión analógica a digital. Mostraremos la elección y sus pruebas.

Capítulo 3.2: Bloque II: este bloque contiene la descripción del sistema que nos permite enviar los datos recogidos al ordenador para posteriormente procesarlos. De igual forma, se describirá en detalle la elección del microcontrolador, el software utilizado para su programación, así como del hardware utilizado para grabar el firmware.

Capítulo 3.3: Bloque III. Es bloque contiene el sistema principal de procesamiento de los datos. Aquí es donde se va a realizar la medición del desfase, así como del registro de los resultados obtenidos en la medición.

Capítulo 4: Este capítulo proporciona una demostración de funcionamiento del equipo, mediante la medida del desfase de un condensador cerámico, al cual se ha aplicado una tensión. Se mostrará los pasos necesarios a seguir para configurar la interfaz de medida y así poder calcular y registrar adecuadamente los datos de desfase.

En las **Conclusiones,** se muestran las diferentes conclusiones y opciones de ampliación del equipo que no se han podido realizar en esta primera versión.

En la **Bibliografía**, se encuentran información de enlace con documentos más completos y detallados, que pueden ayudar a entender más profundamente ciertos aspectos.

En los **Anexos**, se encuentra una gran información como pueden ser una estimación del presupuesto, circuitos detallados, código fuente de los programas, pruebas realizadas de desfase y hojas de catálogo.

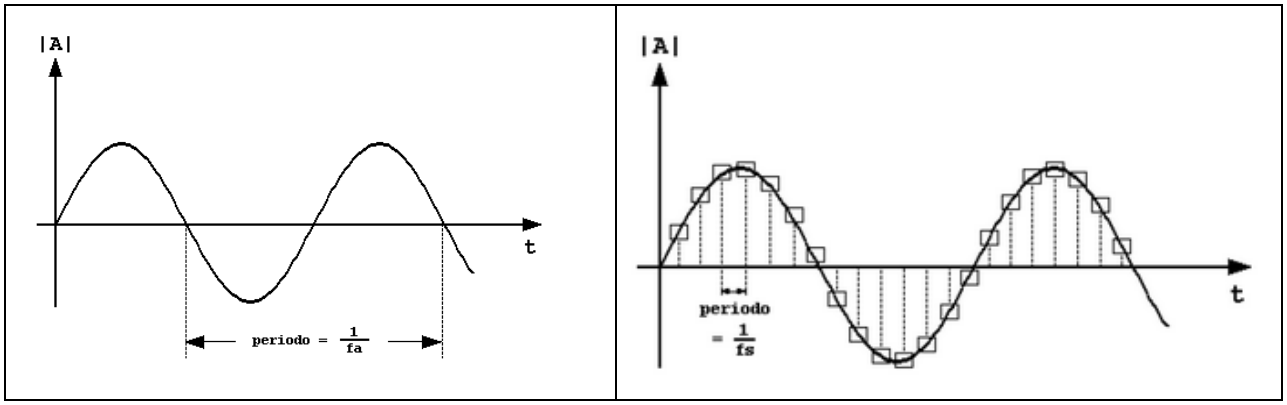
2. Marco teórico

2.1. Teorema de Nyquist

El medidor de desfase de dos señales senoidales debe adquirir los datos de las señales a una frecuencia y muestreo determinado para poder reproducir fielmente la señal analógica pero reconvertida al campo digital.

El teorema de Nyquist nos dice la frecuencia a la que se debe muestrear la señal analógica de entrada para poder ser reconstruida fielmente,

$\mathbf{f_s = 2 \cdot f_a}$	$\mathbf{f_s}$: frecuencia de muestreo
$T_s = T_a/2$	$\mathbf{f_a}$: frecuencia de la señal muestreada (1Hz)

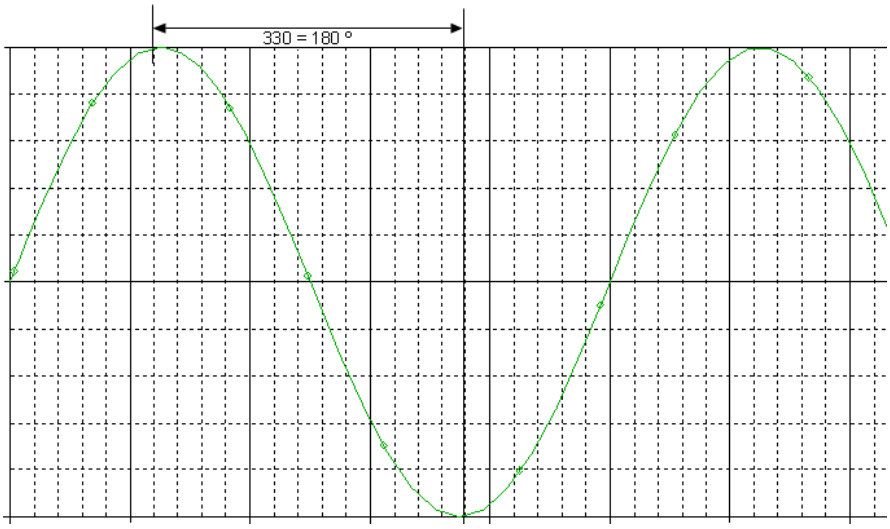


Señal origina y señal muestreada

2.2. Método de cálculo de desfase

La forma de calcular el desfase de dos señales senoidales, se realiza mediante la diferencia temporal al cruce por cero, teniendo en cuenta que las referencias sean de la misma pendiente.

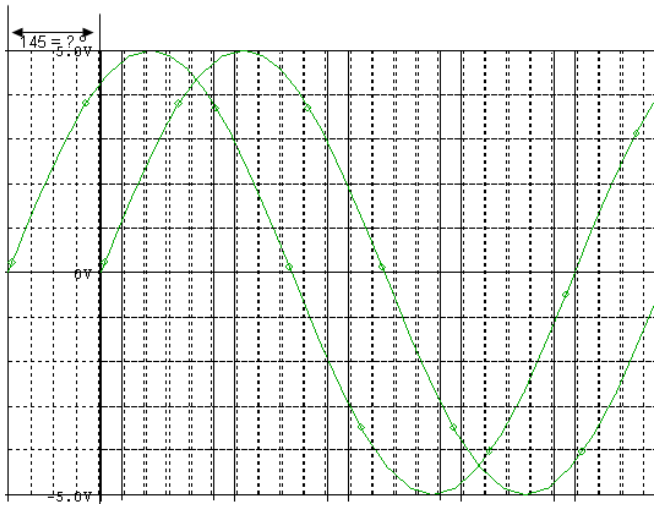
Teniendo una señal senoidal de entrada de 1 Hz, tenemos que tarda en completar medio ciclo 330 ms.



Si tenemos dos señales senoidales de idéntica frecuencia, pero desfasadas en el tiempo, el desfase será la diferencia en el tiempo de cruzar por cero las dos señales.

De esta forma si unimos las dos señales como observamos en la figura y obtenemos que para $\mathbf{y_1}$= 0 e $\mathbf{y_2}$=0 la diferencia en el eje de las x es:

Diff = 145 ms



El desfase será:

$$\varphi = \frac{(Diff * 180)}{K} = \frac{145 * 180}{330} \approx 79°$$

3. Diseño de una interfaz para la medida del desfase de dos señales senoidales.

Fig. 3: Diseño de una interfaz de medida y sistema de adquisición de datos

Previamente al diseño de la interfaz de medida de desfase de dos señales senoidales, se deben adquirir las señales analógicas, por lo que se va a diseñar un sistema de adquisición de esos datos, para un posterior procesamiento mediante la interfaz.

Primeramente se adquirirá la señal senoidal con los ADCs, para después utilizar el microcontrolador que se encargará de hacer un control de los datos de los ADCs y enviarlos a un ordenador que se encargará de procesar los datos.

Si lo comparamos un sistema de adquisición de datos general, las partes comunes con éstos serían la tarjeta DAQ y el ordenador personal.

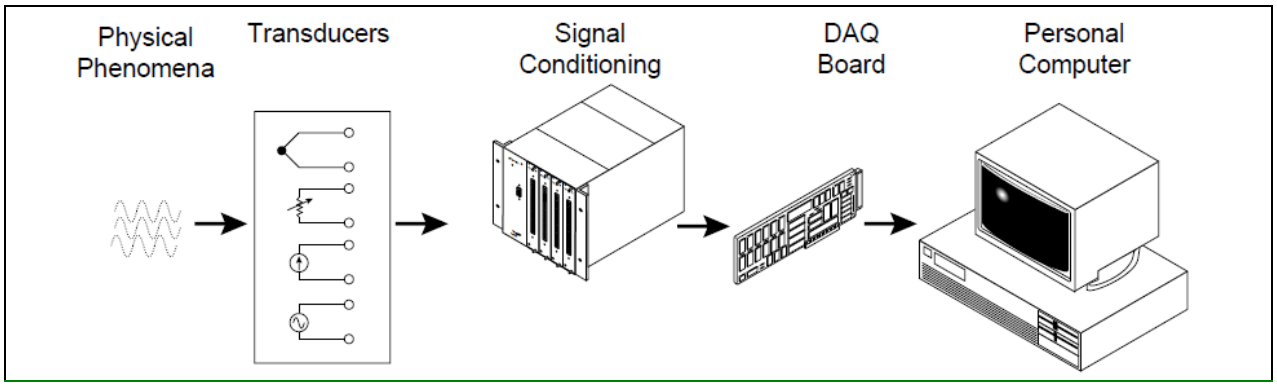
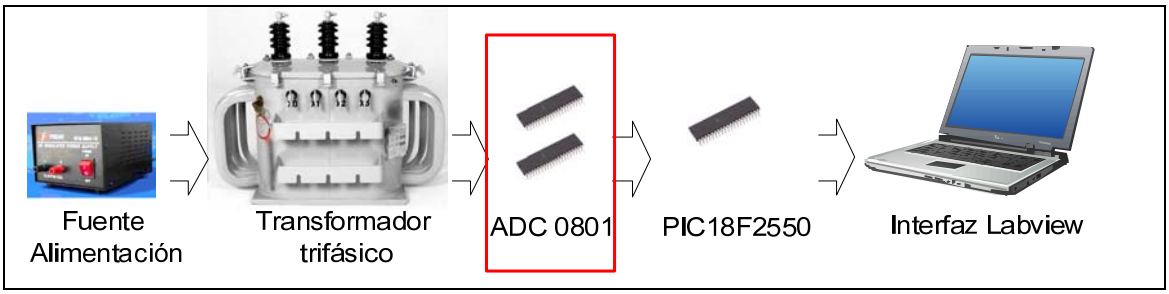


Fig. 4: Esquema de un sistema de adquisición y procesado de datos general

3.1. Bloque I: Adquisición de datos mediante ADC



En esta primera parte de la memoria se va a detallar la adquisición de una señal analógica de un rango 0-5 V y su conversión a una señal digital procesable por el microcontrolador.

Para ello, se debe realizar una selección de un integrado convertidor analógico-digital, que mejor se adapte a nuestras necesidades.

Primeramente se van a analizar brevemente las características más importantes de los convertidores de baja tensión. A continuación, se analizarán las características del convertidor seleccionado. Y finalmente se mostrará la configuración seleccionada para comunicarse con el microcontrolador.

3.1.1. Selección del convertidor analógico-digital

A la hora de seleccionar el conversor A/D, se necesita tener en cuenta una serie de parámetros para poder elegir adecuadamente nuestro dispositivo. Como parámetros básicos tenemos:

- **Tensión analógica de entrada**: máximo valor de tensión de entrada permitido para que el conversor no se sature. Nuestro dispositivo seleccionado acepta una tensión de entrada de 0-5v.
- **Precisión**: máximo error que es posible cometer en una conversión.
- **Estabilidad**: tolerancia a los cambios de temperatura. En nuestro caso no es estrictamente necesario tener en cuenta este parámetro, pues nuestro entorno de funcionamiento no será extremo.
- **Tiempo de conversión**: tiempo que necesita el conversor para proporcionar a la salida el código digital correspondiente a la entrada analógica que se tiene.
- **Formato de salida**: es el formato del código digital: BCD, binario natura, complemento a 1, complemento a 2, etc. En algunos conversores es posible programar el formato de salida.
- **Resolución**: número de bits necesario para llevar a cabo la conversión. En nuestro caso utilizaremos un conversor que nos proporciona una salida de 8 bits, que equivale a 256 valores.

Se han descartado la utilización de los ADC del microcontrolador por ser menos precisos y ser multiplexados. Además el reemplazo de los ADCs, por otros de mejores características o por simple avería es más sencillo y menos costoso.

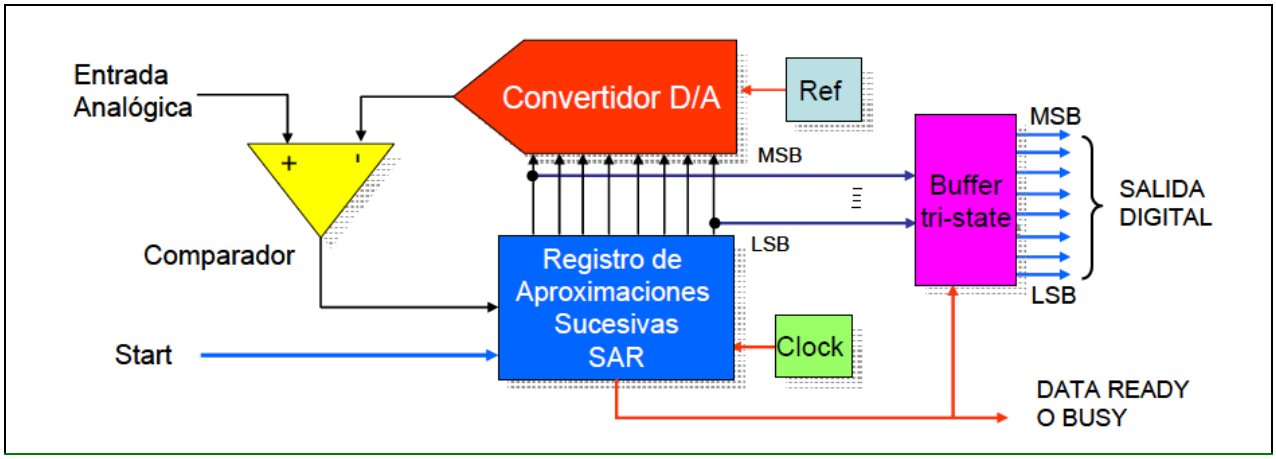
Descripción	ADC integrado (PIC18F2550)	ADC0804
Error de conversión	< 1 LSB	¼ LSB
Tiempo de conversión	<100 µs	100 µs
Resolución	10 bits	8 bits

Tabla 3: Comparativa básica de ADCs

Como se puede observar en la tabla aunque el tiempo de conversión puede ser inferior al ADC, este posee mucho menor error de conversión. Además que nuestro sistema no tiene como prioridad el tiempo, pues nuestra señal de adquisición tiene una frecuencia no superior a 1Hz.

El conversor seleccionado es el modelo ADC0801 de la familia ADC080x del fabricante National Instruments.

El ADC0801 es un conversor CMOS de aproximaciones sucesivas de 8 bits. La entrada diferencial analógica, permite incrementar el Common-Mode rejection ratio de tal forma que reduce el offset de salida a cero, por lo que no requiere de ajuste de cero.



En el diagrama típico de aproximaciones sucesivas, el pulso START inicia el proceso de conversión y deshabilita el buffer tri-state de salida. Al final del periodo de conversión, se activa DATA READY y la salida digital queda disponible en el buffer de salida.

El conversor permite configurar el rango de medida a la entrada hasta 5 voltios, proporcionando a la salida de 8 bits el valor de la conversión en binario natural. El tiempo empleado en la conversión se sitúa en unos 100 µs.

De todos los modelos se ha seleccionado el ADC0801 por ser el que menor error tiene: 1/4LSB, frente al resto con 1/2LSB.

Error Specification (Includes Full-Scale, Zero Error, and Non-Linearity)			
Part Number	Full-Scale Adjusted	$V_{REF}/2$=2.500 V_{DC} (No Adjustments)	$V_{REF}/2$=No Connection (No Adjustments)
ADC0801	±¼ LSB		
ADC0802		±½ LSB	
ADC0803	±½ LSB		
ADC0804		±1 LSB	
ADC0805			±1 LSB

Error del ADC0801 proporcionado por el fabricante

Un conversor que reúne todas las características que requerimos es el ADC0801 y como puntos importantes están:

- Resolución de 8 bits
- Posibilidad de conexión directa al bus del microprocesador
- Tiempo de conversión ≈ 100 μs
- Entrada de voltaje diferencial.
- Entradas y salidas compatibles con ttl
- Generador de reloj dentro del chip.
- Rango de voltaje de entrada de 0v a 5 v. Ideal para utiliza la alimentación USB
- No requiere ajuste de cero.

La resolución con la que vamos ha trabajar es la variación más pequeña de la magnitud que está siendo evaluada y que produce un cambio susceptible de observación en la lectura. La que ofrece la etapa acondicionadora impuesta por el convertidor analógico-digital es la siguiente:

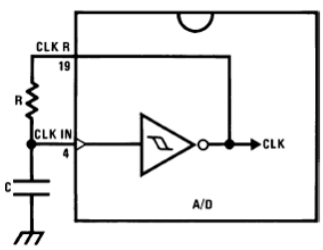
$$\text{Resolución} = \frac{V}{2^n - 1} = \frac{5\,\text{V}}{2^8 - 1} = 19{,}60\,\text{mV}$$

3.1.2. Configuración convertidor ADC0801

El ADC0801 requiere un reloj para funcionar. El reloj puede ser externo conectándolo al terminal CLK IN o puede ser generado por un circuito RC.

El rango de frecuencias del reloj permisibles está entre 100 KHz y 1460 KHz. Para que el tiempo de conversión sea el menor, es conveniente usar la frecuencia más alta posible.

Si el reloj se genera con una red pasiva RC, se utilizan los terminales CLK IN y CLK R conectadas como se muestra en la figura



La frecuencia del reloj se calcula mediante la fórmula:

$$f_{CLK} \cong \frac{1}{1.1\,RC}$$

$$R \cong 10\ k\Omega$$

A Continuación, vamos a ver las señales que se utilizan en el conversor para comunicarse con el microcontrolador:

- $\overline{CS}$: **Chip Select**. Habilita el funcionamiento del convertidor.
- $\overline{WR}$: **Write**. Da la orden de inicio del convertidor.
- $\overline{RD}$: **Read**. Efectúa la lectura de los datos.

Los elementos hardware de configuración de ADC, son muy sencillas. A destacar es la configuración de la tensión de referencia a 2,5 V y la interconexión de la salida de datos ala PIC.

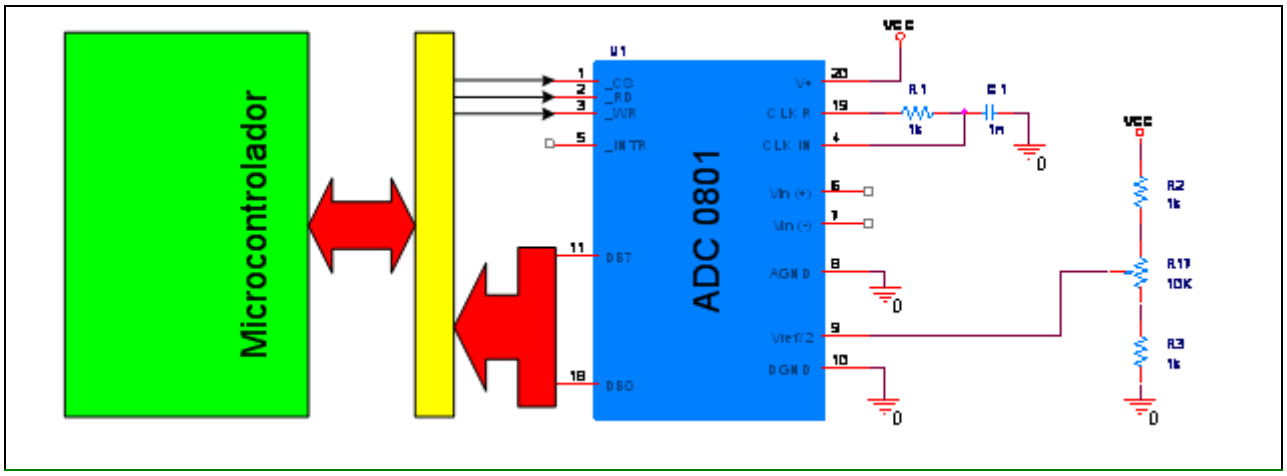


Fig. 5: Interconexión microcontrolador-ADC

Los elementos hardware de configuración de ADC, son muy sencillas. A destacar es la configuración de la tensión de referencia a 2,5 V y la interconexión del bus de datos al microcontrolador.

Se ha decidido no utilizar la patilla INTR del ADC, puesto que no queremos que nos indique el ADC, cuando ha terminado de convertir y por lo tanto este tiempo puede variar, lo cual no nos interesará, por lo que simplemente le pedimos los datos después de un tiempo prudencial.

Según las hojas del fabricante, la lectura de se inicia al activar las señales $\overline{CS}$ y $\overline{WR}$. Con WR y CS con posición lógica "1" el convertidor analógico-digital se bloquea y no actúa.

La conversión empieza con la llegada de un pulso de nivel alto a la entrada de WR siempre que la entrada CS se encuentre a nivel bajo.

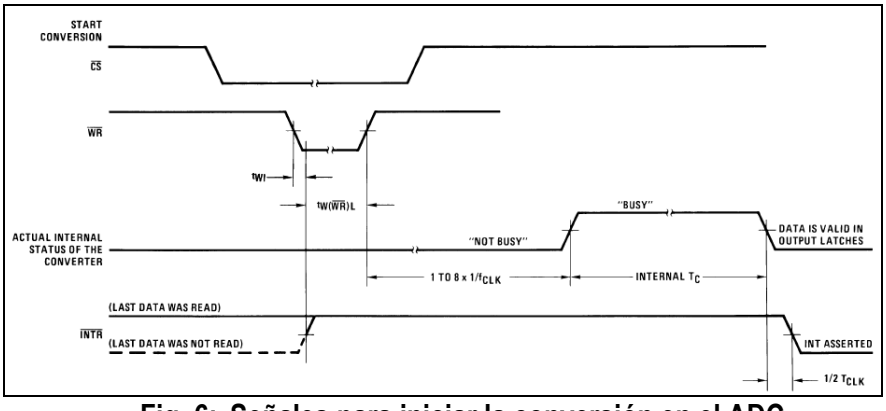


Fig. 6: Señales para iniciar la conversión en el ADC

Durante la transición de nivel alto a bajo de la señal de entrada WR o del CS, el controlador interno se resetea y el registro de datos se ponen a nivel alto "1" lógico.

Cuando el convertidor genera una señal de interrupción INTR, por la pata 5, el byte del dato está listo y puede usarse. Por el motivo descrito anteriormente, este tiempo lo vamos a considerar cerca de 120 μs.

La operación de lectura del RD con CS a nivel bajo habilita los latch de salida, y ya está listo para obtener los datos del ADC.

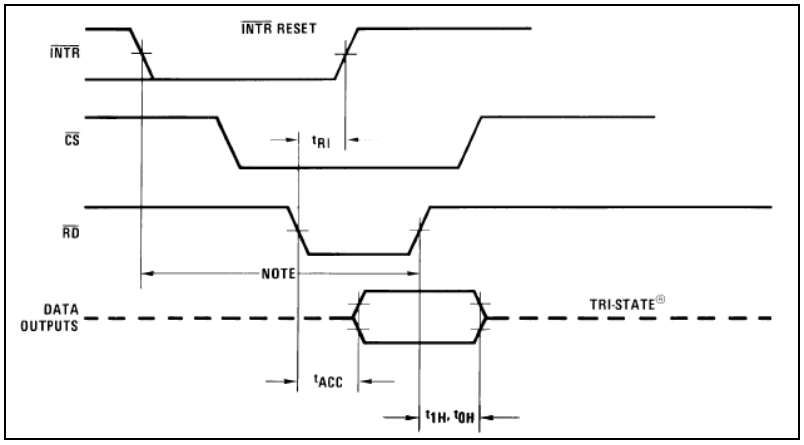
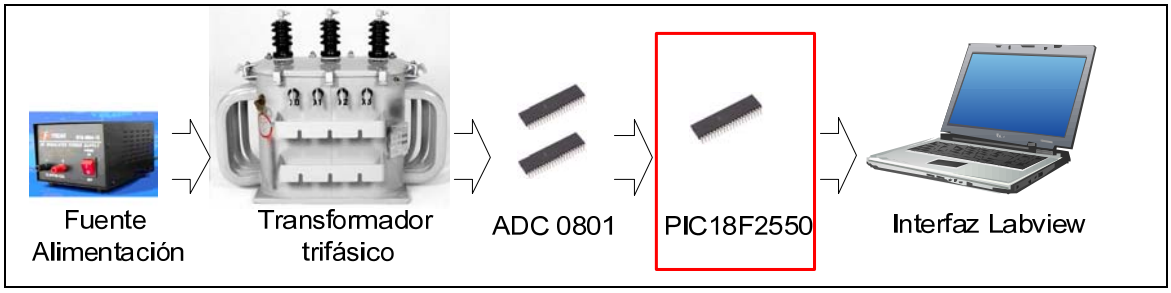
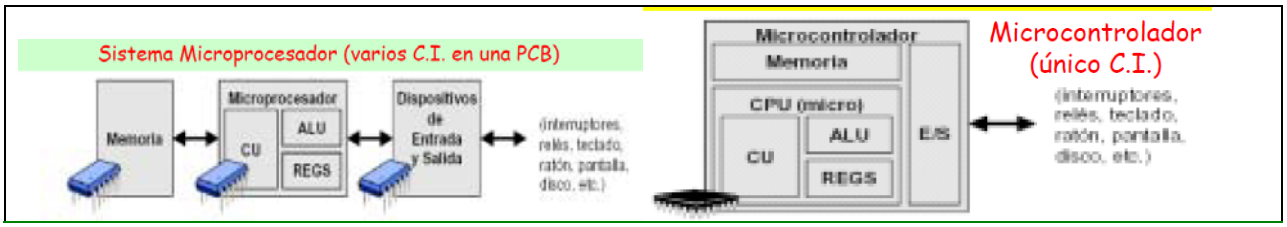


Fig. 7: Señales para recoger los datos del ADC

3.2. Bloque II. Comunicación con PC: PIC18F2550



Para poder transmitir los datos desde un ADC hasta el ordenador, necesitamos de un dispositivo, que proporcione dicho camino. Para ellos, utilizamos un microcontrolador que nos proporcione la suficiente inteligencia como para obtener datos del ADC siempre que lo necesitemos y proporcione un aislamiento de la parte de procesamiento de la información que es el PC.



Diferencias fundamentales entre un microprocesador y un microcontrolador

Un microcontrolador nos proporciona ese dispositivo versátil, de bajo coste y tamaño reducido que nos permite comunicarnos, mediante una comunicación serie con el PC de gestión de la información.

Como vemos en la figura, el microcontrolador nos va proporcionar las características de un microprocesador pero de potencia suficiente para nuestro propósito.

3.2.1. Selección

En el mercado existe una amplia gama de microcontroladores de múltiples fabricante. De entre todos, los más importantes son: Maxim, Microchip, Atmel y Philhips.

Todos los microcontroladores contienen características en común, como puede ser la alimentación a 5V, puertos de comunicaciones (USB, RS-232), puertos de entrada/salida, generación PWM, conversión ADC, amplificadores operaciones, etc., pero pueden diferenciarse de forma muy notable, por la arquitectura interna: RISC o CISC. Los primeros proporcionan un juego de instrucciones muy reducido, mientras que los segundos proporcionan un lenguaje mucho más enriquecido para programar, lo que también reduce las líneas de programación.

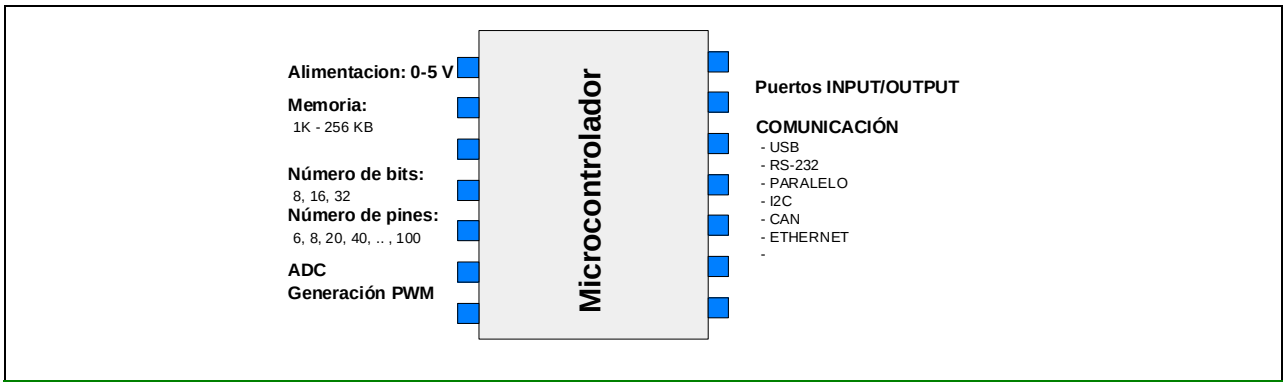


Fig. 8: Características comunes de los microcontroladores

De entre todos los microcontroladores que existen en el mercado, se ha seleccionado como fabricante, Microchip pues contiene una muy amplia variedad de modelos que se ajustan a las necesidades de cada proyecto. Además de que se dispone de gran material y documentación tanto del propio fabricante como a través de la red Internet.

Las características destacadas de los microcontroladores PIC de microchip son las siguientes:

- **Eficiencia** del código: permiten una gran compactación de los programas
- **Rapidez** de ejecución: a frecuencia de 20 MHz es capaz de ejecutar hasta 5 millones de instrucciones por segundo.
- **Seguridad**: Gracias a la separación de la memoria de datos y de programa se asegura el acceso correcto a los registros.
- **Arquitectura RISC** (Juego reducido de instrucciones): esta características de los microcontroladores de Microchip, ha sido una de las grandes ventajas, pues de esta forma se reduce el tiempo de aprendizaje.
- **Compatibilidad** de pines y código: en los dispositivos de la misma familia o incluso de familias distintas posen esta característica.
- Posibilidad de **proteger el código** de forma fiable mediante los fuses.
- **Herramientas**: Como se ha dicho anteriormente se dispone de abundantes desarrollos software, hardware y de muy bajo coste.
- **Encapsulados** de 18 a 20 pines según las necesidades.

De entre todos los dispositivos, se ha buscado uno cuyas características fueran:

- Bajo coste: nos proporcione características avanzadas, pero por un precio bajo.
- Interfaz de comunicación USB: para poder enviar los datos a un dispositivo móvil
- Memoria suficiente: para poder almacenar el programa de control de los datos, programando el firmware en lenguaje "C".
- Al menos 1 puerto de 8 pines: para poder conectar el bus de datos con el convertidor analógico-digital.
- Información de fabricación de programador gratuita.

Device	Program Memory		Data Memory		I/O	10-Bit A/D (ch)	CCP/ECCP (PWM)	SPP	MSSP		EAUSART	Comparators	Timers 8/16-Bit
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)					SPI	Master I^2C™			
PIC18F2455	24K	12288	2048	256	24	10	2/0	No	Y	Y	1	2	1/3
PIC18F2550	32K	16384	2048	256	24	10	2/0	No	Y	Y	1	2	1/3
PIC18F4455	24K	12288	2048	256	35	13	1/1	Yes	Y	Y	1	2	1/3
PIC18F4550	32K	16384	2048	256	35	13	1/1	Yes	Y	Y	1	2	1/3

Tabla 4: Selección de microcontrolador PIC

Hemos seleccionado el dispositivo que posee mayor memoria de datos y programa. El número de entradas y salidas nos es suficiente con 24, pues solo son necesarios 8 bits para la conexión con el dispositivo de adquisición de datos. Y muy importante es la comunicación USB.

Los conversores A/D no nos interesan pues se van a utilizar los ADC externos para comodidad en el recambio de éstos si se estropean.

3.2.2. Diseño del hardware

Como muestra el fabricante del microcontrolador, se debe conectar simplemente un condensador de aproximadamente 470nF a la patilla V_{USB} para el regulador interno de tensión que posee el dispositivo para poder utilizar la transmisión por USB 2.0. De esta forma se evita una circuitería externa que "complicara "el montaje.

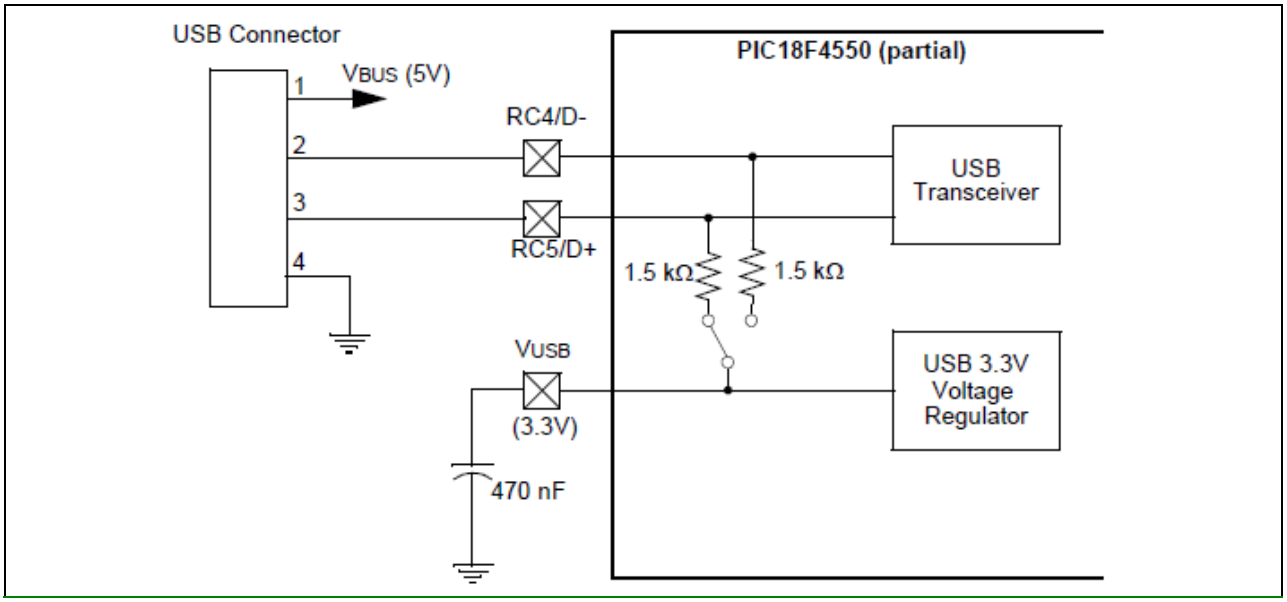


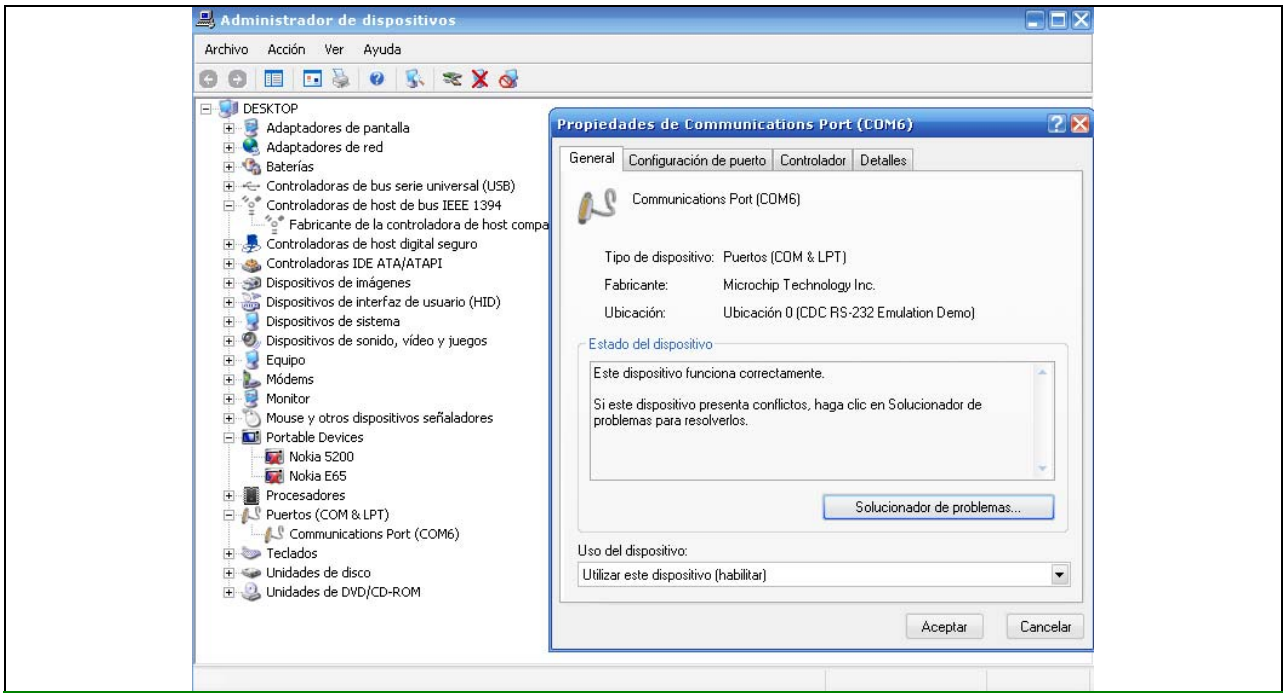
Fig. 9: Esquema USB interno del microcontrolador PIC18F2550

3.2.3. Diseño del software

Para poder utilizar el microcontrolador con el PC es necesario instalar un driver ofrecido por el compilador de C, PCWH, que permite configurar un puesto COM virtual para la conexión USB.

Una vez instalador el driver y conectemos el dispositivo USB, obtendremos el puerto COM virtual al que debemos conectarnos.

Este puerto COM, se configura dentro del administrador de dispositivos de Windows, dentro de los dispositivos COM.

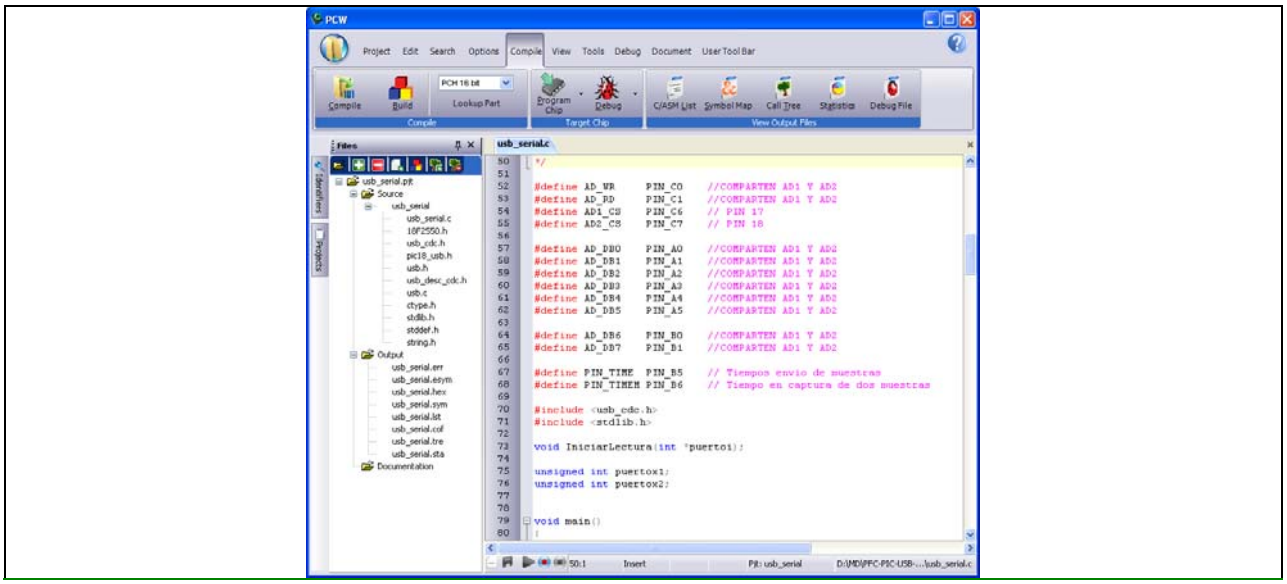


Puerto COM Virtual creado para conexión USB

3.2.4. Programación del firmware en el microcontrolador PIC18F2550

El firmware del microcontrolador se puede programar tanto en C como en ensamblador, pero se ha optado por la primera de ellas, pues reduce el tiempo de análisis, codificación y pruebas.

El software utilizado para la compilación en C, ha sido el de CCS, Inc., pues nos proporciona un entorno de programación muy sencillo e intuitivo, disponiendo de muchas herramientas orientadas a la compilación en C de firmware para microcontroladores.



Entorno de desarrollo CCS PCW

Una vez compilado el código, debemos descargar el archivo .hex al microcontrolador. Para ello se ha fabricado un programador de PICs.

De entre la gran cantidad de opciones de programadores, se ha seleccionado uno muy sencillo que nos permitiese programar el PIC18F2550 y que estuviera disponible el software de forma gratuita.

El programador seleccionado es el ART-2003, que es un programador paralelo y que utiliza muy pocos componentes en su desarrollo. El software de grabación permite programar una amplia gama de microcontroladores entre los que se encuentra el nuestro.

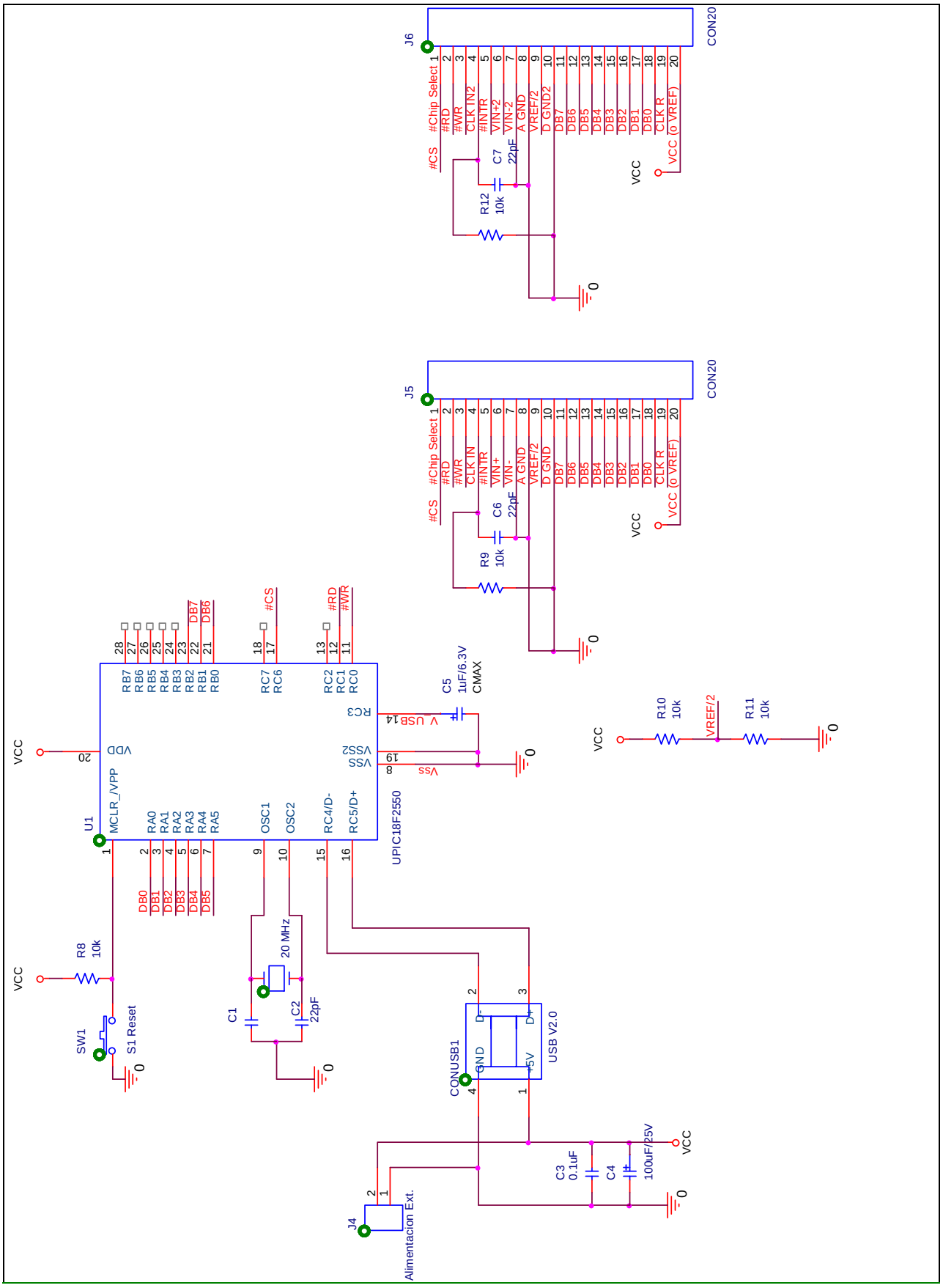


Fig. 10: Circuito de conexión microcontrolador PIC y ADC0801

3.2.5. Diseño del Firmware del microcontrolador

El software introducido en el microcontrolador, primeramente inicializa la configuración USB, y entra en un bucle infinito esperando a que se encuentre una conexión. Una vez que el dispositivo se ha conectado, se analiza si se ha recibido alguno de los caracteres correspondiente a los que tenemos gestionados.

Los caracteres que se gestionan son tres:

- '**B**': si recibimos este carácter, comenzamos a ejecutar la rutina de petición de datos al ADC, según los parámetros configurados.
- '**N**': configuramos la variable ***iMuestras*** que corresponde con la configuración del número de muestras.
- '**T**': configuramos la variable ***iRetraso*** que corresponde con la configuración del tiempo entre muestras.

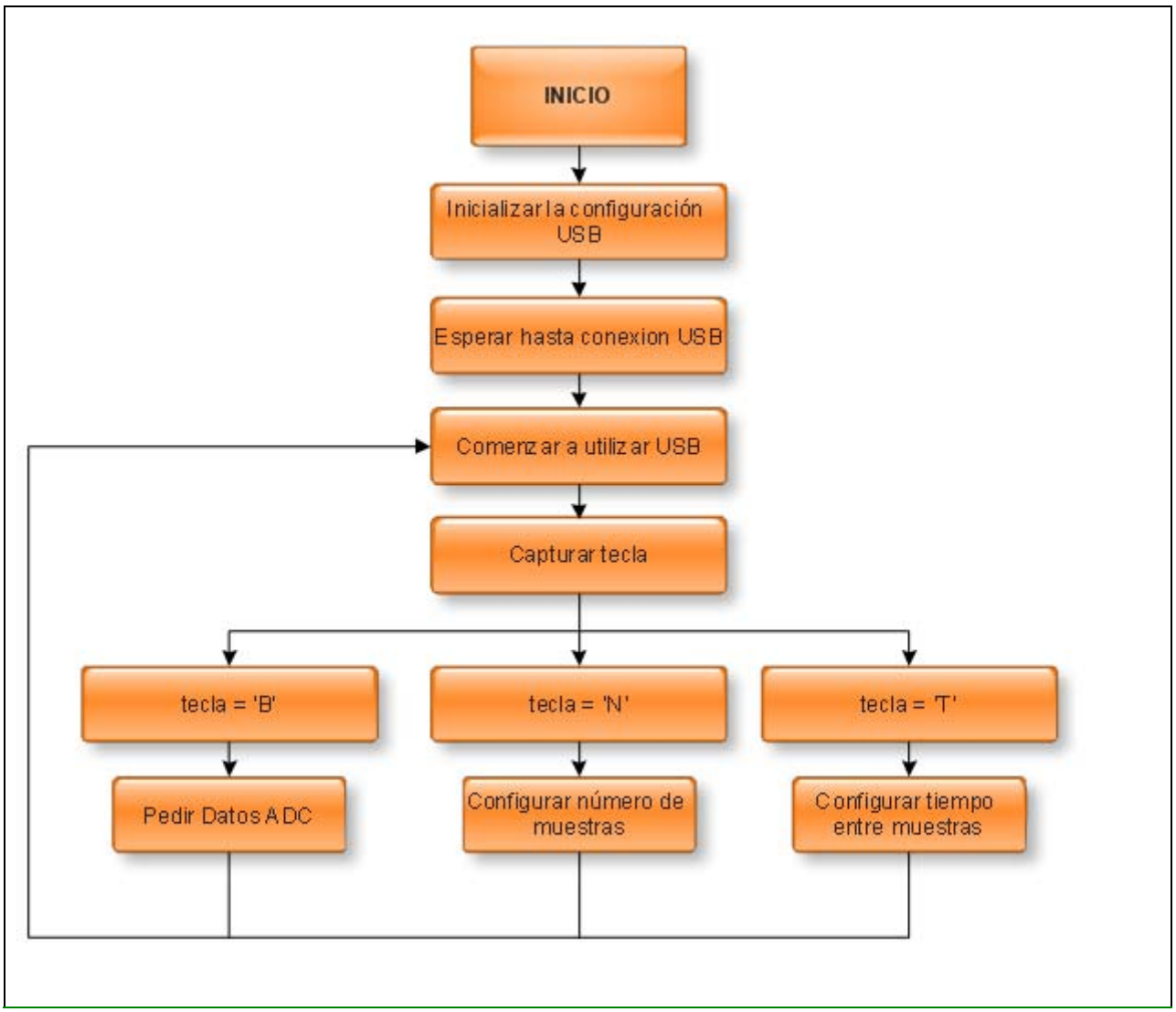


Fig. 11: Diagrama de la rutina principal del PIC18F2550

```
//Inicializamos la configuración USB
usb_cdc_init();
usb_init();

// Entramos en un bucle cuando de dispositivo conectado
while(!usb_cdc_connected()) {}
do {
   // Comenzamos a utilizar USB
   usb_task();

   // Comienza nuestro bucle.
   if (usb_enumerated())
   {
      // Capturamos la tecla enviada
      RcvUSB = usb_cdc_getc();

      // Si se trata de una 'B', estamos utilizando el ADC externo
      if (RcvUSB=='B') ...

      // Configuracion de mustras y tiempo de muestreo
      //################################################
      if (RcvUSB=='N') ...

      if (RcvUSB=='T') ...

      //continue;
      }

   }while (TRUE);
```

Extracto código: Programa principal

A continuación se muestra un extracto del código que corresponde con la configuración del número de muestras. Como se puede ver, una vez que hemos recibido una 'N', volvemos a leer la entrada con la función ***usb_cdc_getc()***, esperando encontrar el carácter 'C', con lo que vamos guardando carácter a carácter que va formado el número, una vez obtenido los datos en un array de caracteres, lo convertimos a número y lo introducimos en nuestra variable de control de número de muestras ***iMuestras***.

Por último enviamos un mensaje de respuesta al PC, mediante la función ***printf***(), con el número de muestras configurado.

```
if (RcvUSB=='N')
{
   RcvUSB = usb_cdc_getc();
   if (RcvUSB=='C')
   {
      szDatos[0] = usb_cdc_getc();
      szDatos[1] = usb_cdc_getc();
      szDatos[2] = usb_cdc_getc();
      szDatos[3] = usb_cdc_getc();
      iMuestras = atol(szDatos);

   }

   printf(usb_cdc_putc, "\nConfigurado con: \n  - Numero de muestras %Lu.", iMuestras);
   continue;
}
```

Extracto de código: Configuración del número de muestras

El siguiente diagrama muestra la petición de datos a los ADCs, según la configuración.

Como se puede ver se entra en un bucle cuya repetición está condicionada por el número de muestras que debemos enviar.

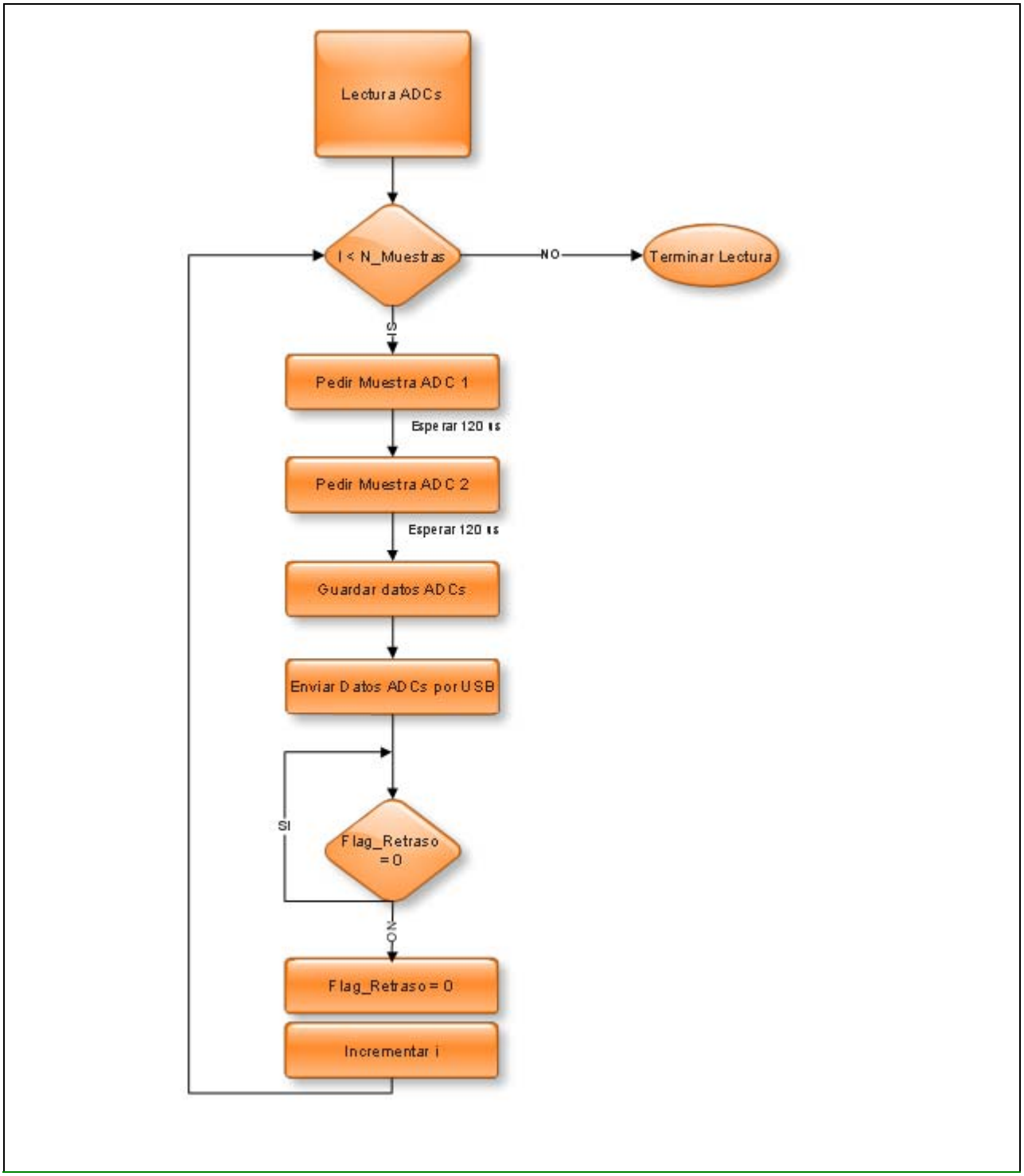


Fig. 12: Diagrama de Bloques: Petición de datos a los ADCs

Primeramente, leemos del primer conversor, para lo que primero habilitamos el ADC activando la patilla Chip Select a nivel bajo. Entonces, provocamos un cambio de pendiente de H a L, y después de esperar un tiempo de 120 μs, procedemos a la lectura de los datos del conversor. A continuación hacemos lo mismo con el segundo conversor y enviamos los datos al PC.

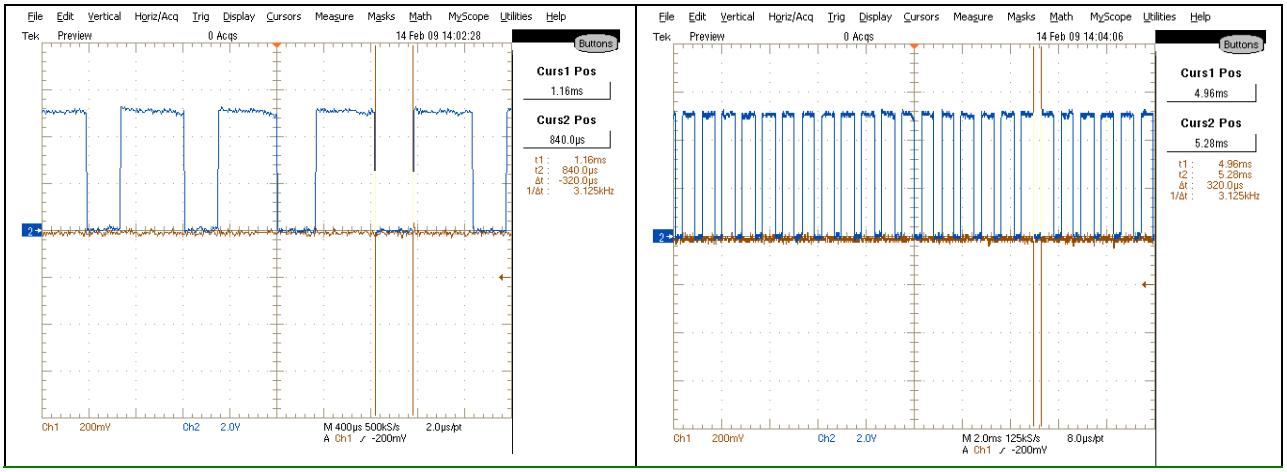
El extracto de código se puede ver la llamada a la rutina ***IniciarLectura***, a la que le pasamos como parámetro una variable donde queremos que nos guarde el contenido de la lectura.

```
// Si se trata de una 'B', estamos utilizando el ADC externo
if (RcvUSB=='B')
{
   for (i=0; i<iMuestras; i++)
   {
      output_low(PIN_TIMEM);  // PIN TEST

      // Desctivamos ADC2 y Activamos ADC1
      output_high(AD2_CS);
      output_low(AD1_CS);

      output_high(AD_WR);
      output_high(AD_RD);
      output_low(AD_WR);    // Inicial Lectura y esperar
      output_high(AD_WR);   // Inicial Lectura y esperar
      delay_us(120);
      IniciarLectura(&puertoa);

      // Desctivamos ADC1 y Activamos ADC2
      output_high(AD1_CS);
      output_low(AD2_CS);

      output_high(AD_WR);
      output_high(AD_RD);

      output_low(AD_WR);  // Inicial Lectura y esperar
      output_high(AD_WR); // Inicial Lectura y esperar
      delay_us(120);
      IniciarLectura(&puertob);

      // Desctivamos ADC1 y ADC2
      output_high(AD1_CS);
      output_high(AD2_CS);

      // Guardamos los datos a enviar y reseteamos las variables
      puertox1=puertoa;
      puertox2=puertob;
      puertoa=puertob=0;

      // Enviamos los datos
      printf(usb_cdc_putc, "%03u", puertox1);
      printf(usb_cdc_putc, "%03u", puertox2);


      output_high(PIN_TIMEM); // Pin de Test

      // Esperamos el tiempo establecido
      // Sumamos iRetras+tiempo de conversion (PIN_TIMEM)
      delay_us(iRetraso);
   }
   // Fin for que guarda datos en el array
```

Extracto de código: Petición de datos a los ADCs

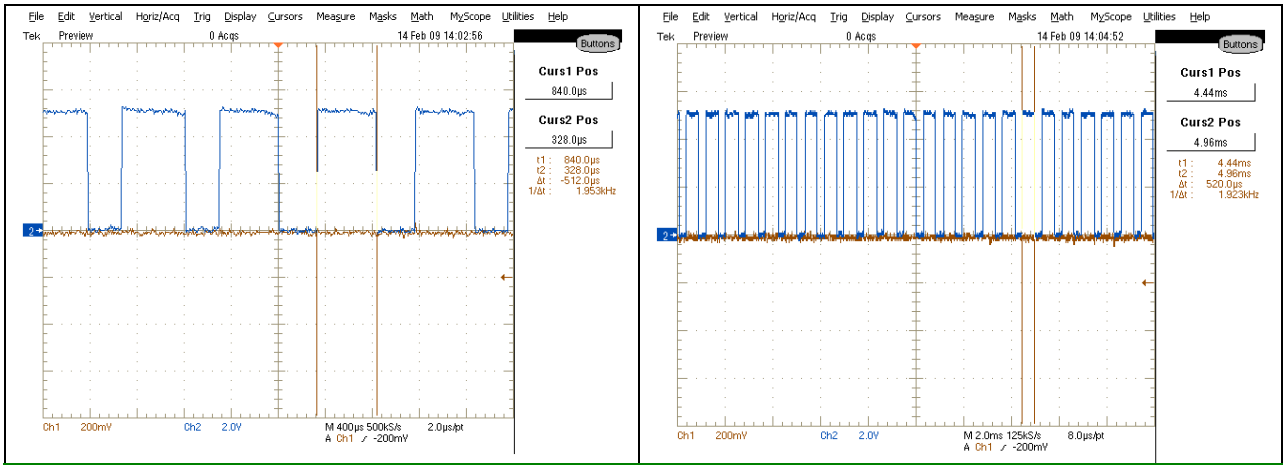
3.2.6. Tiempos de muestreo

El tiempo total de obtención de una muestra de cada ADC y envío a la interfaz es de aproximadamente 320 µs. Este tiempo es medido mediante el pin 6 del puerto B, que se ha habilitado para pruebas. Como se puede observar el tiempo es prácticamente constante.



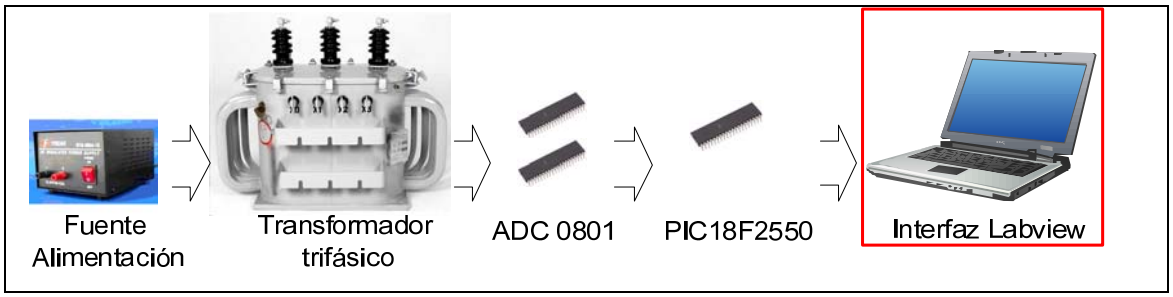
Tiempo de recogida de datos

En la siguiente gráfica se muestra el tiempo de espera entre una muestra y la siguiente que ha sido configurado a 512 µs. En este tiempo el microcontrolador no hace ninguna tarea y se queda esperando que transcurra el tiempo configurado.



Tiempo de espera entre muestras

3.3. Bloque III: Representación y registro de datos. Interfaz de medida en Labview.

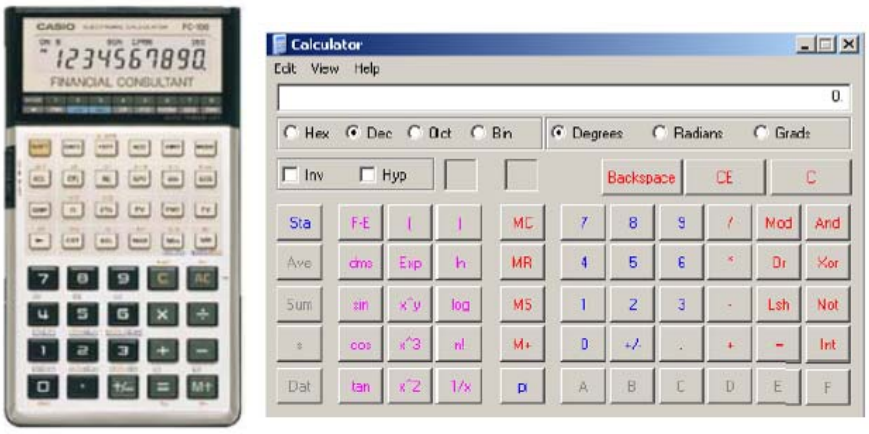


Para registrar y procesar los datos obtenidos correctamente, necesitamos de una interfaz que se encargue de la gestión de todos esos datos.

La interfaz de comunicación, nos permite además registrar los datos para un posterior análisis si se desea, con otros programas.

National Instruments ofrece un software de procesamiento de información muy completo llegando a nivel SCADA. En nuestro caso, deseamos obtener los datos, procesarlos y obtener más datos.

LabVIEW es la abreviación para Laboratory Virtual Instrument Engineering Workbench. Es una poderosa y flexible herramienta para la adquisición, análisis y presentación de datos que puede calificarse como un entorno excelente para aplicaciones de instrumentación y control de procesos. En este proyecto se utiliza la versión profesional 8.5. Este software nos permite desarrollar, entre otras cosas, instrumentos virtuales con una interfaz gráfica muy amigable, permitiendo que cualquier persona con conocimientos en la materia para la cual se desarrolla el instrumento, pueda interactuar y al mismo tiempo cambiar y/o ajustar los campos variables en el instrumento como si éste existiera físicamente



Instrumentación virtual: Una calculador real... una calculadora virtual

3.3.1. Diseño de la interfaz de medida en Labview

Nuestra interfaz de medida se ha divido para su uso como se muestra en el siguiente diagrama de bloques.

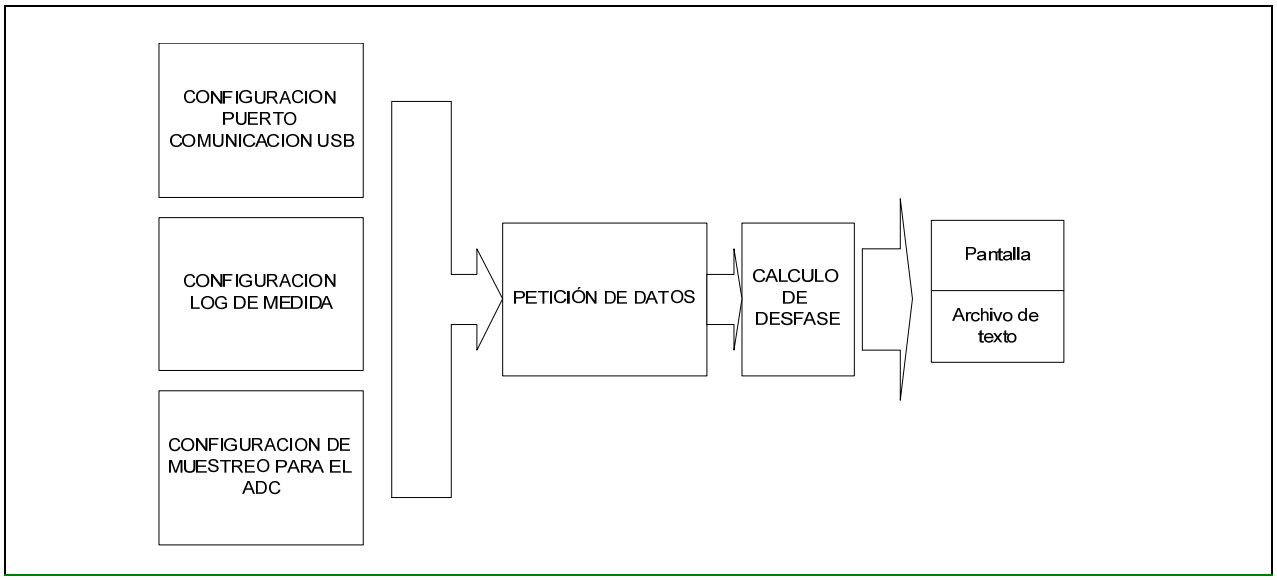


Fig. 13: Funcionamiento Interfaz de cálculo de desfase

Como se muestra en el diagrama, previamente a la obtención de datos, debemos configurar el puerto virtual, el tiempo de log, el tipo de muestreo. Y una vez hecho esto realizaremos la petición de los datos, una vez obtenidos todos los datos, y procesados, calcularemos el desfase, cuyo resultado se puede presentar en pantalla con en un documento de texto.

Para poder crear un interfaz completo de medida del desfase, se ha tenido en cuenta las siguientes especificaciones:

- Sistema que se pueda configurar el número de muestras y tiempo de muestreo para poder medir diferente frecuencias.
- Sistema que muestre las señales adquiridas visualmente.
- Sistema de registro de datos para un posterior procesamiento.
- Sistema autónomo de medida.

Los datos recibidos del microcontrolador PIC se guardan en dos arrays que representan los valores convertidos a digital de cada ADC, por lo que éstos son las dos estructuras de las que se van a partir para aplicar el método de calculo de desfase.

3.3.2. Módulos de la interfaz

En esta parte se van a describir brevemente los módulos que componen la interfaz, y que van desde dicha recepción de datos del PC hasta la visualización del resultado del cálculo de desfase.

3.3.2.1. Paneles de configuración de muestreo y envío de comandos

El siguiente panel, permite configurar el microcontrolador con el tiempo de muestreo y el número de muestras que deseamos adquirir, y de esta forma configurarlo para varias frecuencias.

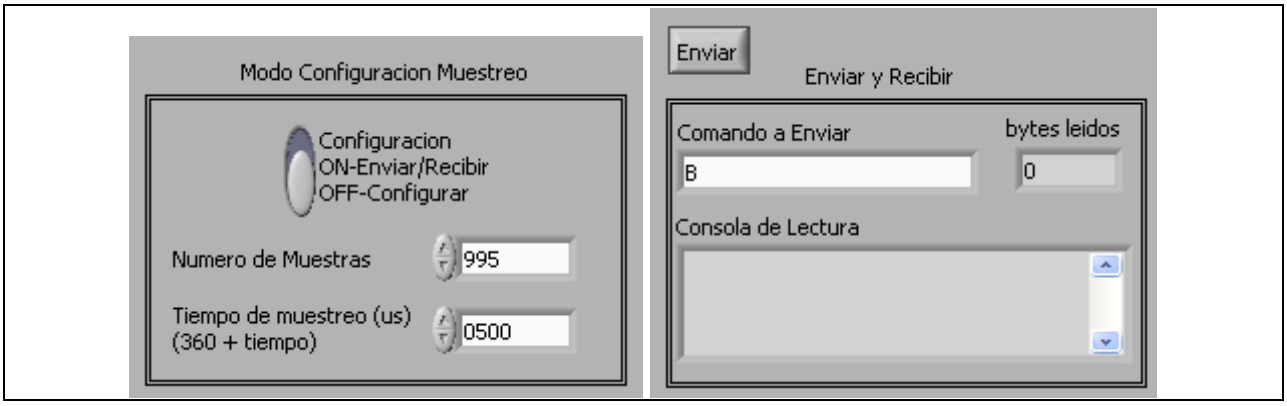


Fig. 14: Paneles de configuración del muestreo y envío de comandos.

Primero se debe configurar los parámetros y enviar la información al microcontrolador, éste responderá con la nueva configuración.

Seguidamente se debe activar el interruptor a modo ON-Enviar/Recibir y enviar el comando **B**, para adquirí ir los datos de la señal.

3.3.2.2. Panel de configuración del puerto virtual

Mediante el siguiente panel se configura el puerto de comunicaciones. Previamente se debe instalar el controlador gratuito de puerto virtual de CSS PIC, que proporciona un puerto tipo COM pero con características de puerto USB.

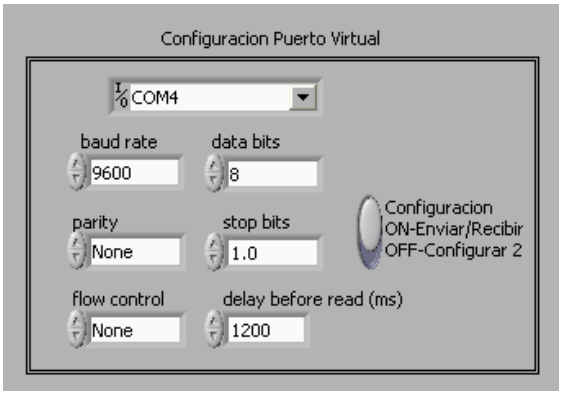


Fig. 15: Panel de configuración de puerto virtual

3.3.2.3. Panel de configuración de registro de datos.

En este panel se puede configurar el intervalo de tiempo de obtención de muestras. También podremos acceder directamente al archivo de texto con el log de las medidas.

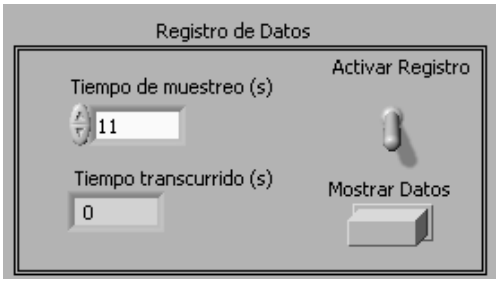


Fig. 16: Panel de configuración de registro de datos

Al iniciar la interfaz de medición, se debe especificar el archivo donde se quiere registrar los datos de salida de registro de desfase.

La imagen que se muestra a continuación representa este archivo en el que los datos que se guardan tienen el formato:

DESFASE	FRECUENCIA DE MUESTREO	NUMERO DE MUESTRA	INTERVALO DE TIEMPO DE RECOGIDA

```
LabVIEW Measurement
Writer_Version  0.92
Reader_Version  1
Separator       Tab
Multi_Headings  No
X_Columns       No
Time_Pref       Absolute
Operator        elCarras
Description     Datos de Medida de desfase
Date    2009/12/14
Time    14:06:27,96875
***End of Header***

Channels        1
Samples 1
Date    2009/12/14
Time    14:06:28,03125
X_Dimension     Time
X0      0.00000000000000000E+0
Delta_X 1.000000
***End_of_Header***
X_Value Untitled        Comment
        0.000000        Grado de desfase. Frecuencia de muestreo:  500. m/s Numero de muestras: 1000. Intervalo de recogida:     5 s.
        229.000000      Grado de desfase. Frecuencia de muestreo:  500. m/s Numero de muestras: 1000. Intervalo de recogida:     5 s.
        159.000000      Grado de desfase. Frecuencia de muestreo:  500. m/s Numero de muestras:  995. Intervalo de recogida:     5 s.
        282.000000      Grado de desfase. Frecuencia de muestreo:  500. m/s Numero de muestras:  995. Intervalo de recogida:     5 s.
```

Archivo de salida del registro de datos

3.3.2.4. Panel de visualización de señal de entrada

Aquí podremos ver la señal muestreada a partir de los datos introducidos en los arrays. El tiempo es orientativo y de momento no se ha especificado ninguna escala en particular.

También se tiene la posibilidad de **ajustar el offset** de la señal de entrada, para poder obtener el desfase de una forma más precisa, y para poder calcular el desfase adecuadamente.

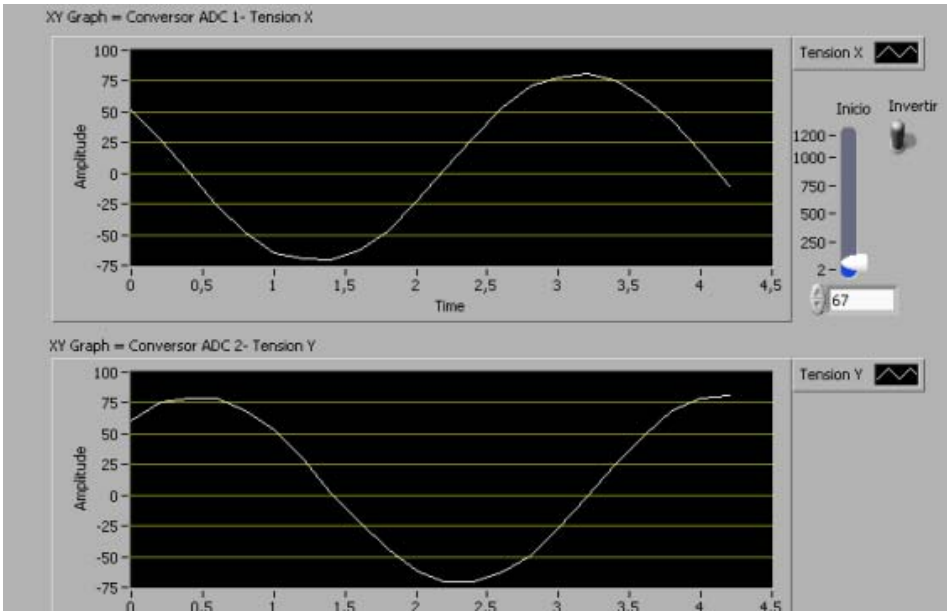


Fig. 17: Panel de visualización de señales de entrada

3.3.2.5. Panel de calculo de constante K

Este panel visualiza las dos señales muestreadas y con ella se puede calcular manualmente la constante K para medio ciclo de onda. También se pueden visualizar todos los valores de entrada. Como se describirá más adelante este valor es necesario configurarlo, para poder tener como referencia como es la señal (frecuencia) y por tanto poder calcular correctamente el desfase.

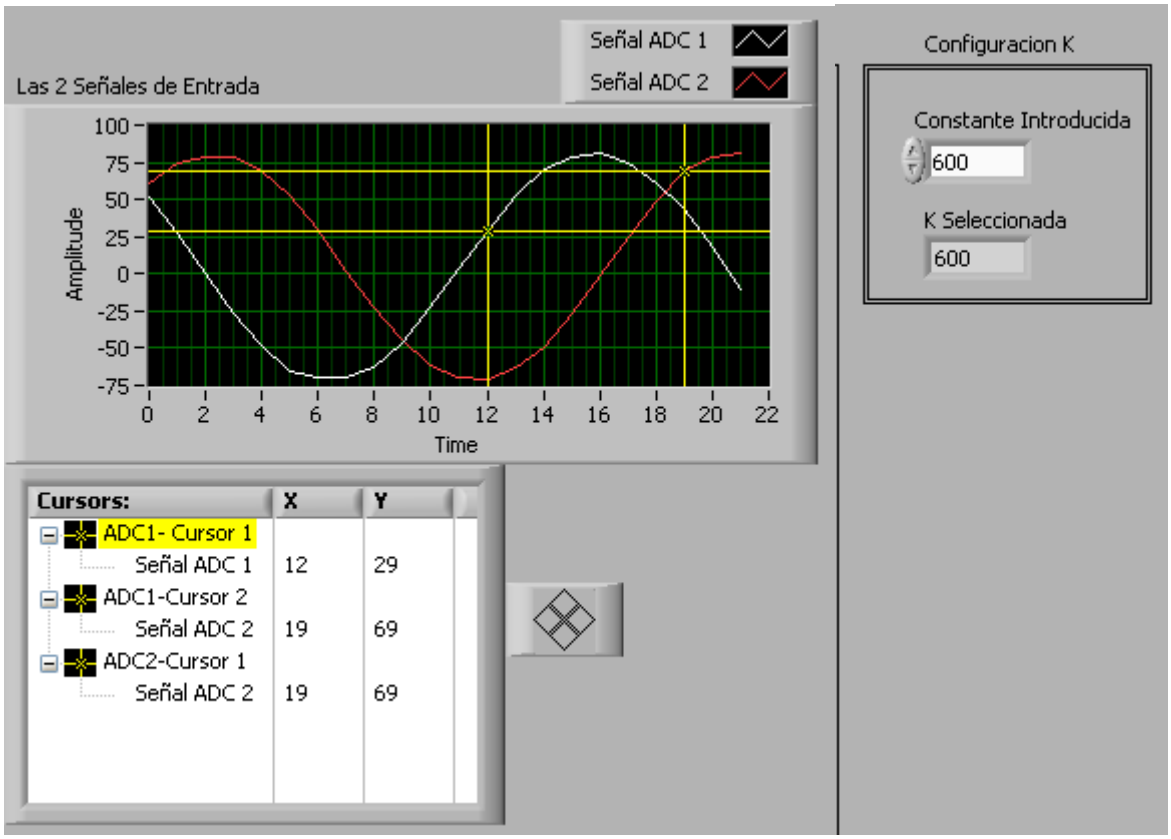


Fig. 18: Panel de cálculo de constante K

3.3.2.6. Panel resultado

Este panel muestra información del desfase obtenido. Además se muestra información del elemento encontrado para el cálculo del desfase, así como los índices de la posición donde se ha encontrado en cada array de datos de cada ADC, el elemento buscado.

Fig. 19: Panel resultado

3.3.2.7. SubInstrumento virtual

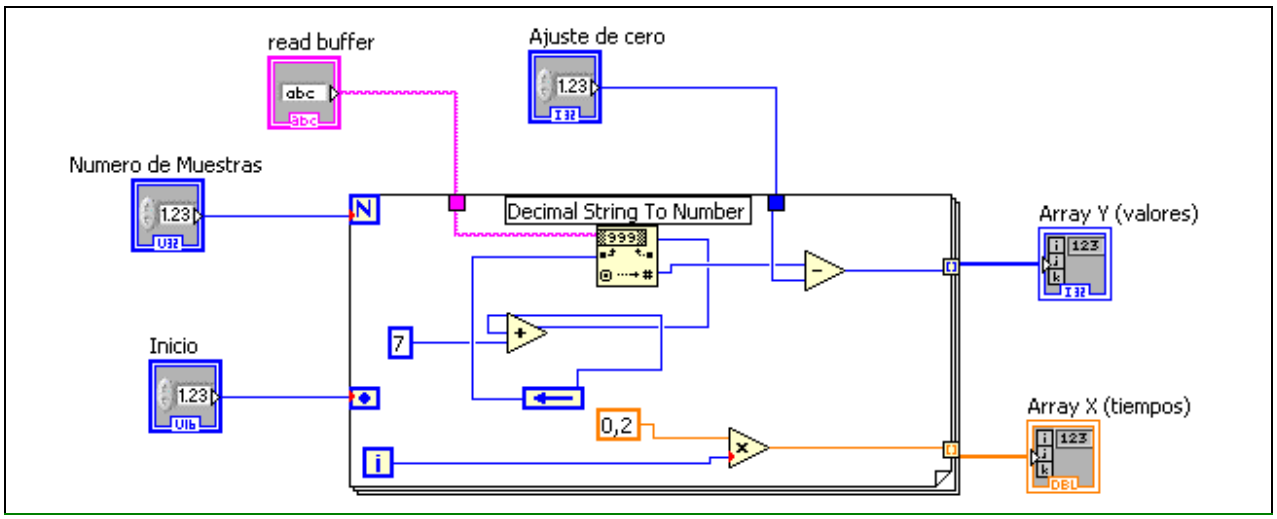
A continuación, se muestra un subintrumento virtual creado para obtener los datos de entrada y guardarlos en un array, el cual se utilizará para el cálculo del desfase.

Bloque constructor de array

Este bloque permite construir dos array de datos a partir de los datos de entrada que envía el microcontrolador.

Un array corresponde a los datos del eje de las x y el segundo array se forma a partir de los datos de entrada. Por defecto se ha puesto como incremento 0,2 en el eje x, mientras que el eje y contiene los datos del ADC.

Array X (tiempos)	0	0,2	0,4	0,6	0,8	1	1,2	1,4
Array Y (valores)	43	42	41	41	39	38	37	37



Interior del subinstrumento virtual: Bloque constructor de array

3.3.3. Cálculo desfase

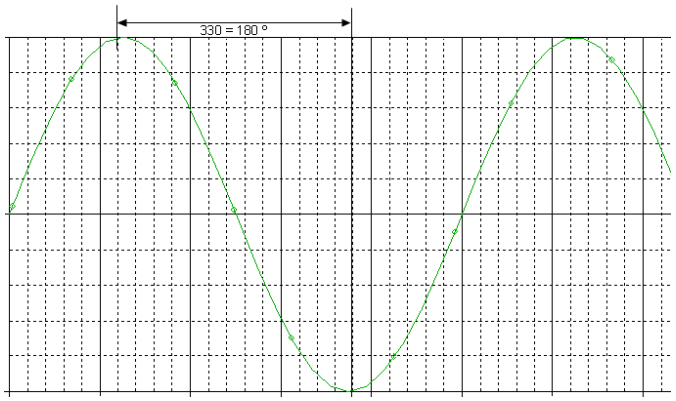
Teoría

Como se ha comentado en la previamente, para calcular el desfase tendremos que tener una referencia del tiempo que dura el semiciclo de la onda, es decir, necesitamos conocer la frecuencia de la señal senoidal y cuanto ocupa en nuestro contexto, los arrays.

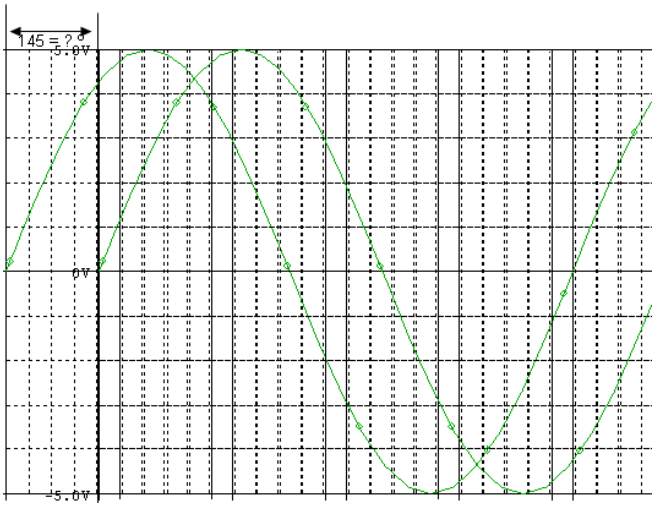
Para el cálculo del desfase inicialmente, se debe configurar la constante K, que representa el valor de la señal senoidal de medio ciclo.

Array Y (valores)								
Índice i	0	1	…	**30**	31	..	**360**	361
Valor almacenado	0	10	…	**128**	127	..	**-128**	-120
	v_1= 128 v_2 = -128 K= i_1-i_2 = 360- 30= **330**							

Así por ejemplo en la figura adjunta, tenemos que el valor de nuestra constante K tiene valor 330, que corresponde con el valor del array de datos de entrada.



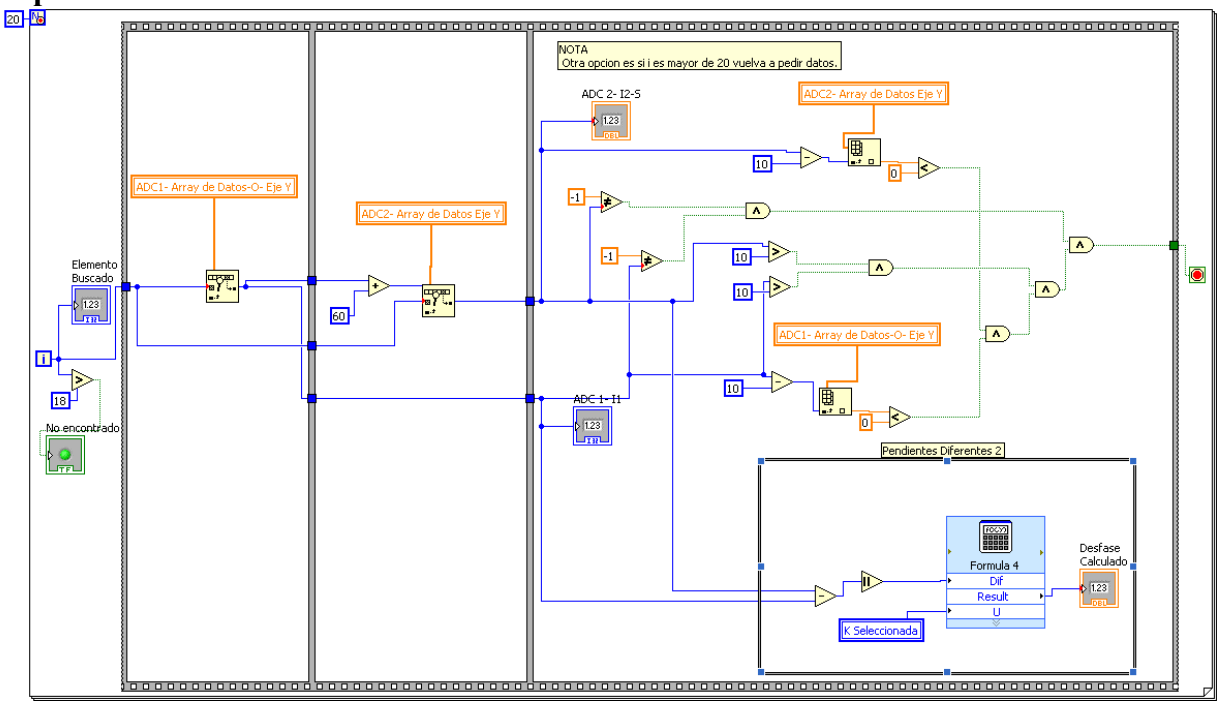
De esta forma si unimos las dos señales y obtenemos para $\mathbf{y_1}$= 0 e $\mathbf{y_2}$=0 la diferencia de x: Diff = 145.



El desfase será:

$$\varphi = \frac{(Diff * 180)}{K} = \frac{145 * 180}{330} \approx 79°$$

Implementación en Labview del calculo de desfase



La imagen muestra el algoritmo de cálculo del desfase, que aunque parece bastante complejo visualmente, es un calculo sencillo y se acompaña del diagrama de bloques que hay más abajo para explicarlo.

Para implementar este cálculo en Labview, debemos, primeramente **buscar el valor** para el cual $y_1(x) = 0$ en los dos arrays, siempre que se cumpla que su pendiente sea la misma (es decir misma referencia), teniendo en cuenta como referencia primera de paso por cero el array de elementos del ADC1.

Una vez encontrados estos valores x_i, podremos aplicar la fórmula y calcular el desfase.

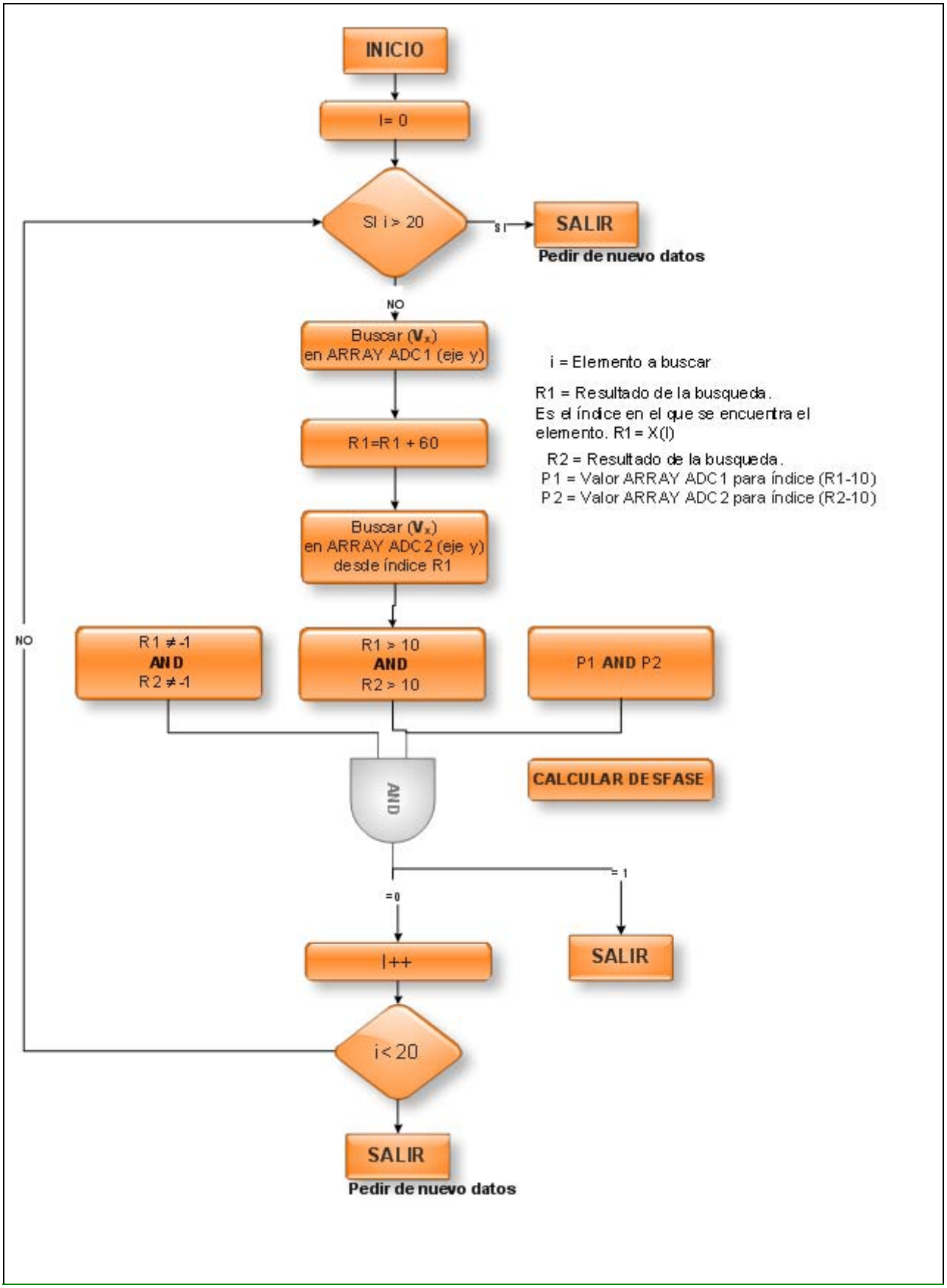
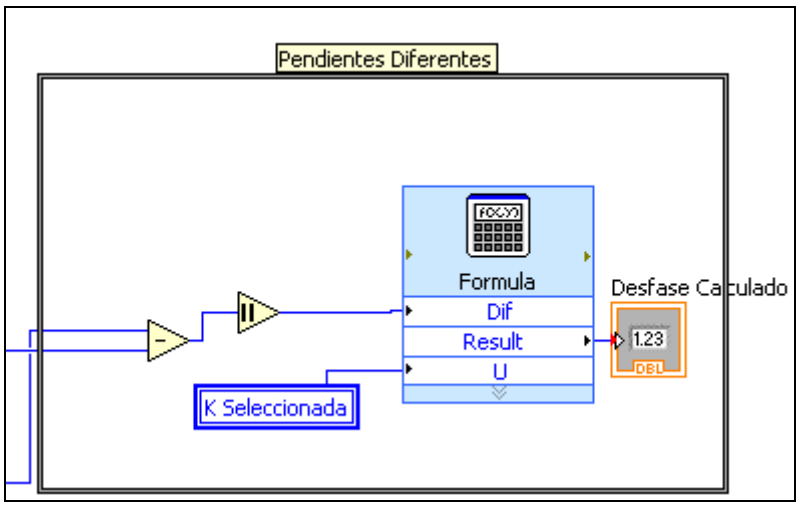


Fig. 20: Diagrama de bloques de implementación del cálculo de desfase en Labview

Las condiciones, para poder saber si se trata de la misma referencia son las siguientes:

- R1 = -1 . Si el resultado de la búsqueda es igual a -1, significa que no se ha encontrado el elemento en el array.
- R1 > 10. Esto es un valor, que hemos puesto como referencia, para que se empiece a buscar el elemento a partir de 10 posiciones más en el segundo array.
- P1 AND P2. Con esta condición definimos que estamos haciendo cálculos de la misma pendiente. Es decir tenemos los mismos valores de referencia.

Si todos los valores dan TRUE, el valor de desfase calculado será el correcto y es calculado mediante el bloque siguiente:



Cuya fórmula es la utilizada para calcular el desfase:

Desfase ϕ = (Dif*180)/U

4. Configuración y funcionamiento del medidor de desfase.

La medición del desfase conlleva primeramente la configuración el medidor con unos valores iniciales necesarios para poder realizar la medida.

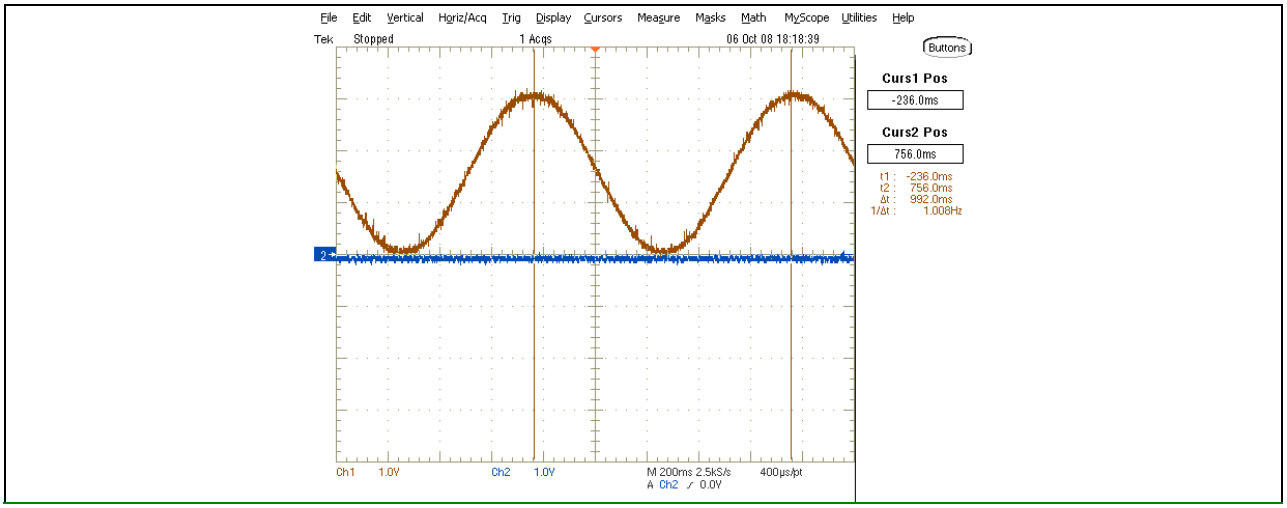
Como muestra el diagrama primeramente, deberemos saber la frecuencia de las señales de entrada, por ello habrá que ajustar los valore de muestreo.

A continuación es necesario medir una constante K con las herramientas de la interfaz de medida, que represente el valor de media fase de la onda.

Una vez configurado si deseamos el registro de los datos calculados, ya se podrá iniciar la medida.

Fig. 21: Pasos de configuración para la medida del desfase

Como aplicación del medidor de desfase, se va a mostrar el proceso de trabajo para medir el desfase introducido por un condensador de 8,2 nF al que se aplica una tensión senoidal de 1 Hz y 3Vpp.



Señal Senoidal aplicada a un condensador

La señal aplicada al condensador nos servirá mediante un divisor de tensión como primera señal de entrada a uno de nuestros terminales de medidor.

La señal de salida del condensador, será convertida de corriente a tensión mediante un aparato que realiza esta conversión reduciendo el ruido. Después de acondicionarla eliminado el ruido, ajustando la amplitud y el offset, será la segunda señal de entrada del medidor de desfase.

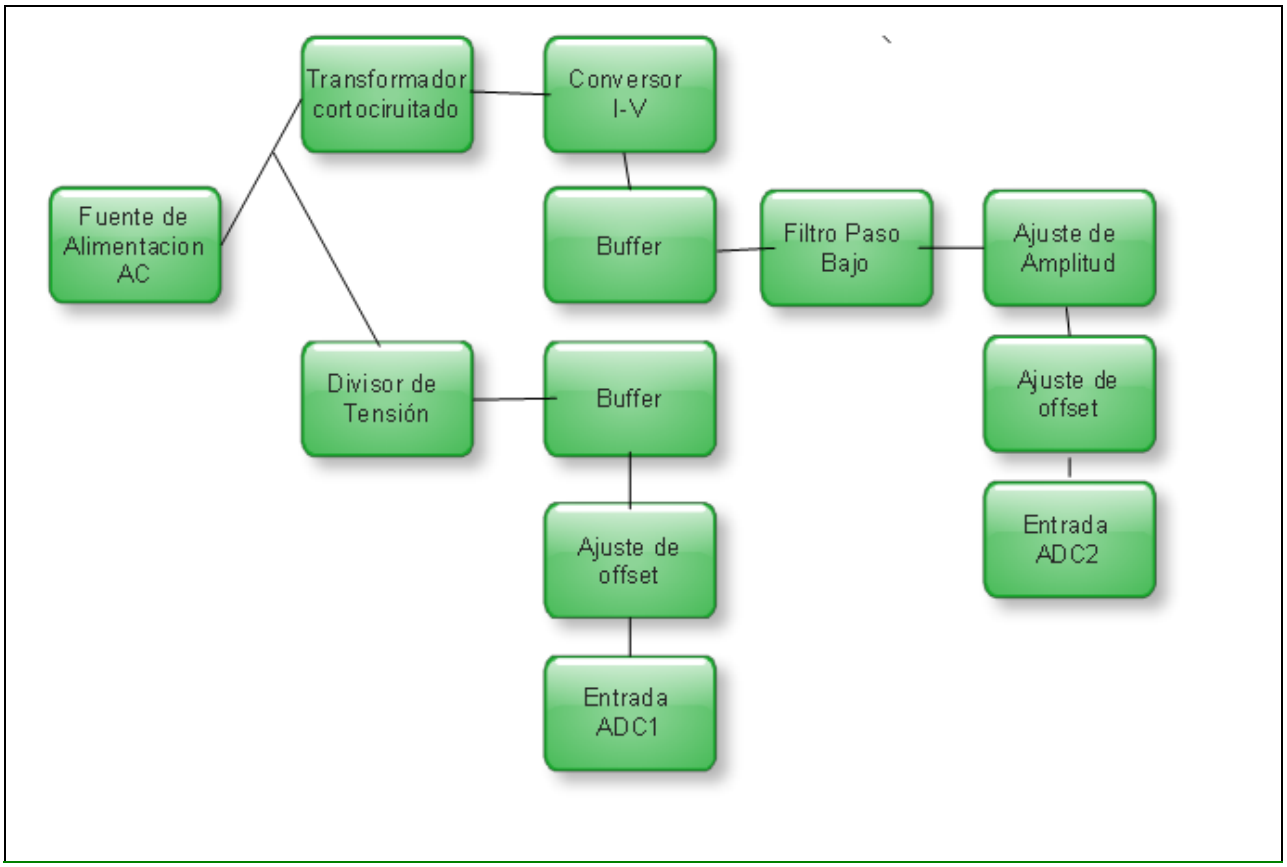
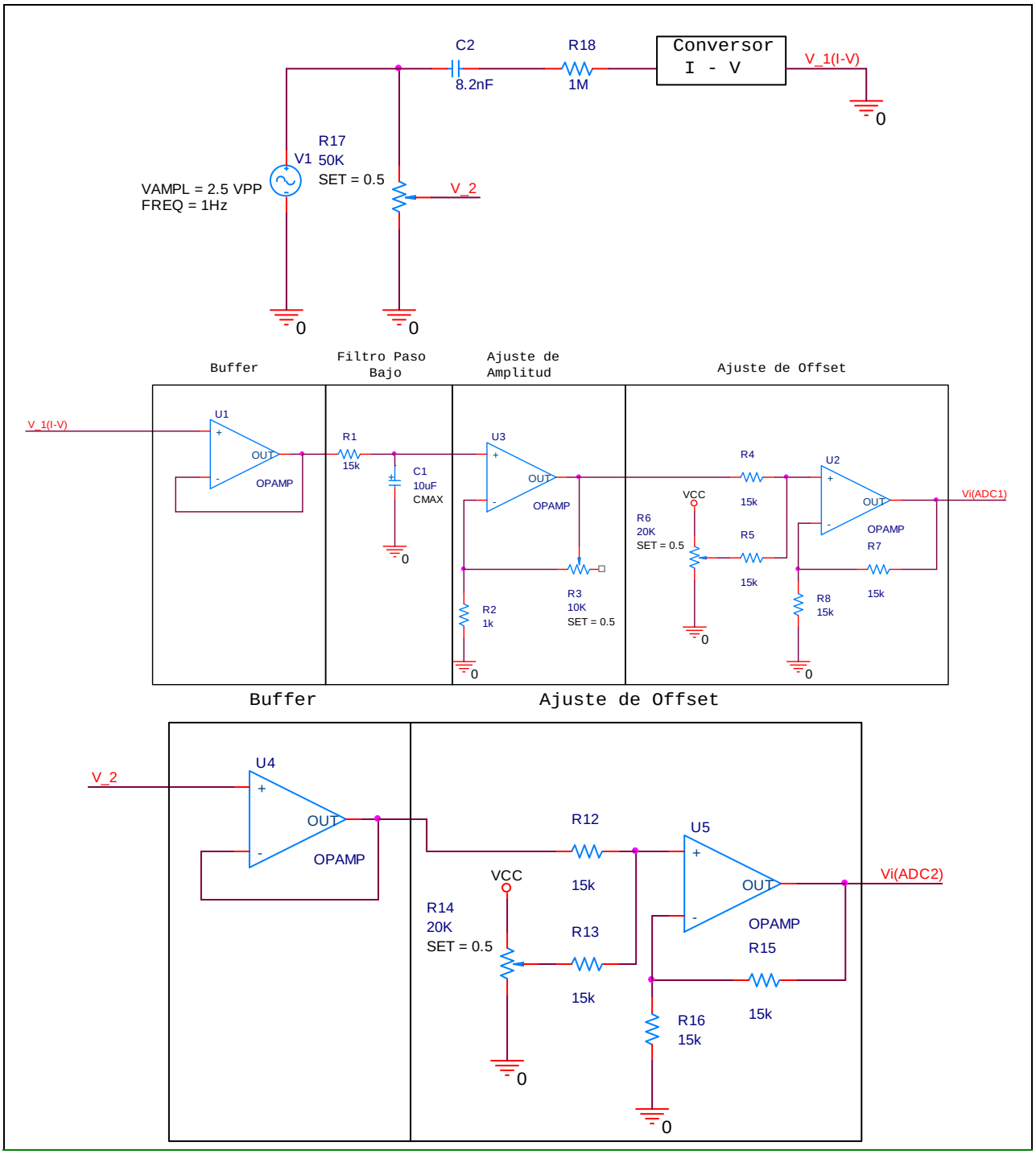


Fig. 22: Diagrama de bloques del circuito propuesto para medir el desfase de un condensador

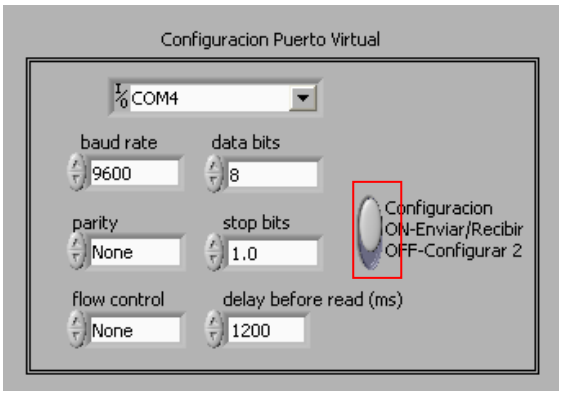
A continuación se muestra el circuito eléctrico correspondiente al diagrama de bloques anterior, con el que se ha acondicionado la señal que se ha introducido a través del condensador de 8,2 nF.



Esquema eléctrico del circuito acondicionador

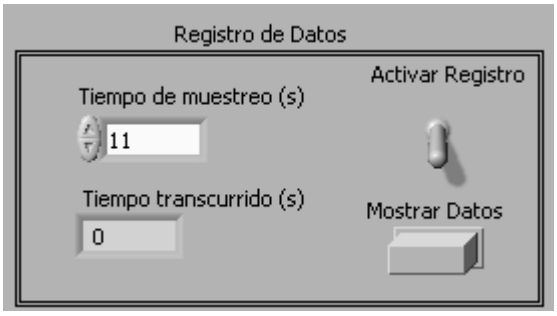
Paso 1- Configuración del puerto virtual

Primeramente, se debe seleccionar el puerto virtual donde se ha configurado el dispositivo USB. Y pasar a modo Normal, una vez configurado.



Paso 2- Configuración del registro de datos

En este panel configuramos el tiempo de registro de datos. Una vez activado en la parte inferior, se muestra el tiempo que lleva transcurrido después de la última toma de datos.



Paso 3- Configuración de la frecuencia de trabajo

Esta etapa es importante a la hora de realizar la medida, pues si se configura incorrectamente, el muestreo puede ser no suficiente (submuestreo) y por tanto no se reconstruirá correctamente la señal.

El tiempo de configuración está en µs por lo que se necesita sumar además los 300 µs, que tarda en adquirir los datos del ADC. Así para la señal que presentamos a continuación es:

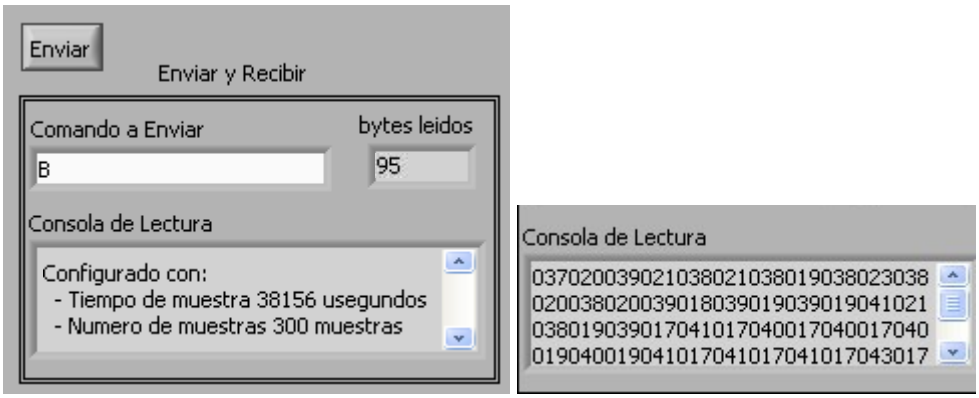
Frecuencia señal entrada: 1 Hz

$$\mathbf{f_t \geq 2 \cdot 1\ Hz = 2\ Hz \rightarrow t_t \leq 500\ ms/muestra}$$

Si muestreamos a 500+300 µs / muestra, y multiplicamos por el número de muestras, obtenemos:

800 µs/m · 995 muestras = 796 ms → solo obtenemos una parte de la señal, que es la que nos interesa.
796 ms /1000 ms * 100 = 79,6 % de la señal obtenida.

Una vez introducidos los datos, se debe pulsar el botón enviar y obtendremos un mensaje de respuesta en la consola, confirmando la configuración introducida.

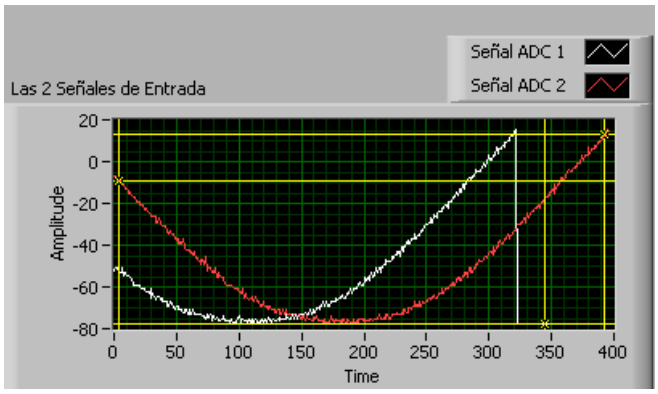


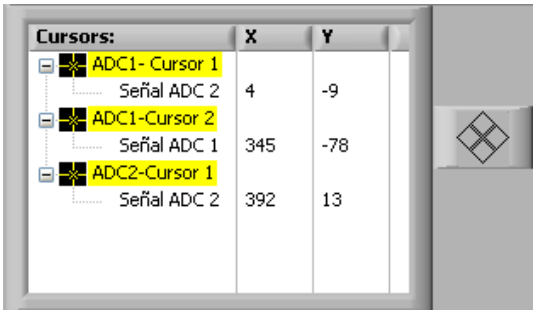
Paso 4- Configurar la K

Para poder calcular debemos tomar una primera muestra, para ello debemos activar el cursor, y ya podemos obtener la muestra:

El sistema necesita utilizar una constante que depende de la frecuencia de la señal que se quiere muestrear. Por ello, se debe configurar correctamente esta constante. Esta constante representa el número de filas del array que representa medio ciclo.

Para calcular, se pueden utilizar los cursores de la gráfica conjunta. Se mira la diferencia en el eje de las x para el cual y es igual, y esa es el valor de K.



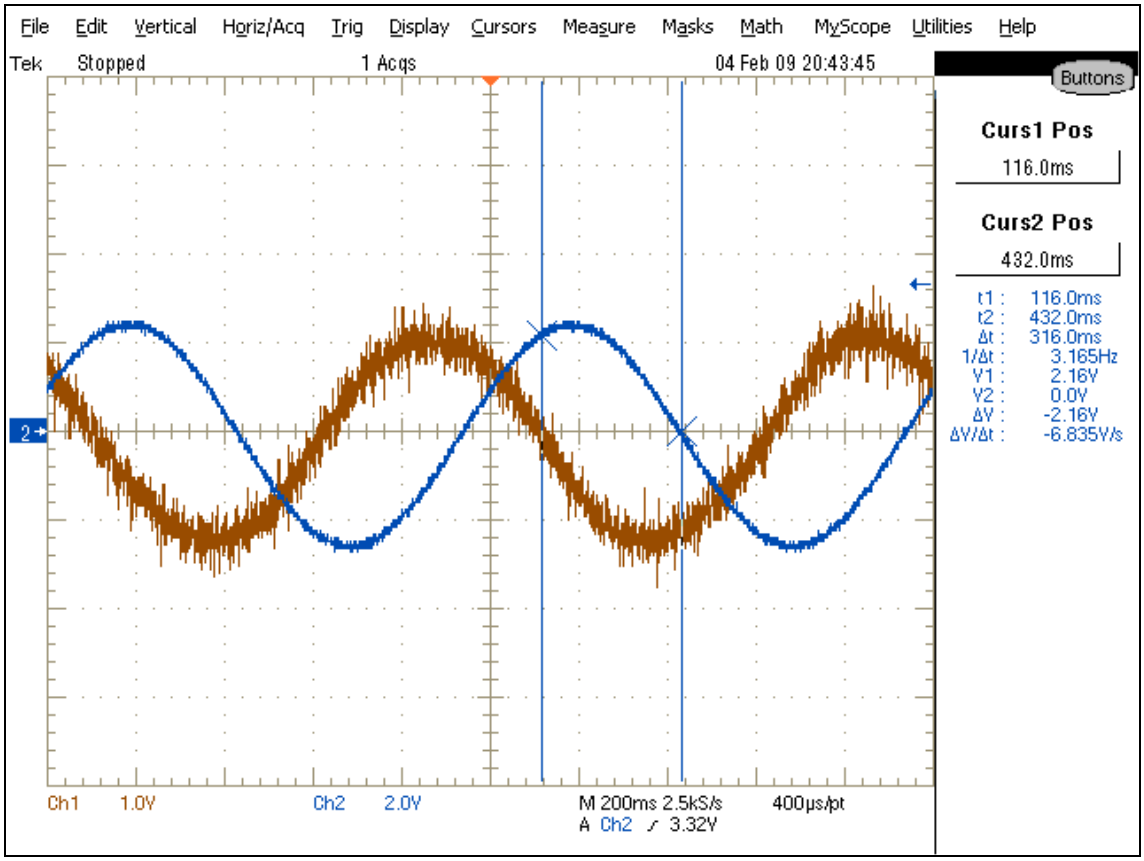


En la imagen que se muestra arriba, vemos que restando 392-4 = 388, que es el valor en unidades del array de Labview que corresponde a medio ciclo de la señal.

Paso 5- Calcular el desfase

Observamos que en la consola obtenemos datos de muestras, Y ya automáticamente, se piden muestras para calcular automáticamente el desfase, obteniendo el siguiente resultado:

Si el resultado lo comparamos con el valor que muestra el osciloscopio, podemos ver que el dato de desfase es muy similar al real:



Datos de desfase del Osciloscopio

Si calculamos el desfase con los datos del osciloscopio, tenemos:

desfase ϕ = 306º* 316ms / 1000 ms = **113, 7** º

En este caso el desfase se introduce el aparato conversor de corriente tension.

5. Conclusiones

El desarrollo de la interfaz de medida de desfase es bastante complejo pues se compone de muchas partes, sin embargo el cálculo de desfase en sí obedece a una regla muy sencilla y fácil de implementar.

De todo ello se observar que donde se ha tenido mayores dificultades ha sido la adquisición de los datos de entrada e integración con la interfaz.

El medidor de desfase desarrollado intenta tener las características de un equipo ya existente en el mercado llamado PAM360, pero con una interfaz de medida y registro de datos. El equipo existente tiene como objeto la medición de relés de protección y la utilización en test de transformadores.

Nuestro equipo tiene como objetivo medir el desfase de dos señales senoidales para poder obtener la tg δ, que representa el deterioro de los transformadores como se ha comentado en la introducción. Para esta tarea existe un equipo más completo que es el IDA200.

Por tanto, este proyecto servirá como base, para realizar un sistema mucho más completo y fiable con características añadidas.

Como futuras ampliaciones se han propuesto las siguientes:

- **Pantalla LCD**: un display LCD que muestre directamente el valor del desfase calculado.
- **Microcontrolador** con funciones matemáticas: están normalizándose en el mercado microcontroladores con funciones matemáticas como el dspic de Microchip. Mediante un microcontrolador con esta tecnología se podría realizar los cálculos del desfase directamente desde él y mostrar directamente la salida en una LCD.
- **ADC**: podría usarse modelos más precisos.
- **Memoria RAM** o incluso **memoria flash por usb**: así se podrían guardar los datos de desfase calculado, sin necesidad de tener un ordenador, y cada cierto tiempo volcar los datos a un pendrive USB. Esto haría ganar el equipo en portabilidad.

PAM360™	IDA200™
	
· Mide el desfase para ensayar relés de protección direccional y para efectuar ensayos direccionales en transformadores del instrumental de medida.	· Diagnostica material de aislamiento en la mayoría de los objetos de una subestación, como transformadores, transformadores para la medida, casquillos, cables aislados con papel, etc.

Equipos existentes en el mercado, con objetivos parecidos a nuestro proyecto

6. Bibliografía

- [Web] CCS PCW. Compilador C para PIC
  - Web: http://www.ccsinfo.com/
- [Web] Foros de Labview
  - Web: http://www.labview.com
- [Web] Primeros Pasos con LabVIEW:
  - Web: http://cnx.org/content/col10592/latest/
- [Web] Fuente de conocimiento de Nationoal Instrument (Knowledge base)
  - Web: http://sine.ni.com/kb/
- [Web] Toda la información sobre el protocolo USB.
  - Web: http://www.usb.org
- [PDF] Datasheet Medido de Desfase PAM 360
  - Archivo: pam360_es.pdf
- [PDF] Datasheet microcontrolador PIC18F2550
  - Archivo: pic18f2550.pdf
- [PFC] Proyecto fin de carrera: Medido de distancia por ultrasonidos
  - Autor: Javier Sáez Cardador
- [Web] Fabricante de micrcontroladores: Maxim, Atmel
  - Web: http://www.maxim-ic.com/
  - Web: http://www.atmel.com/

Apéndice A. Presupuesto

El coste total del proyecto, se puede dividir como muestra el diagrama en tres partes diferenciadas.

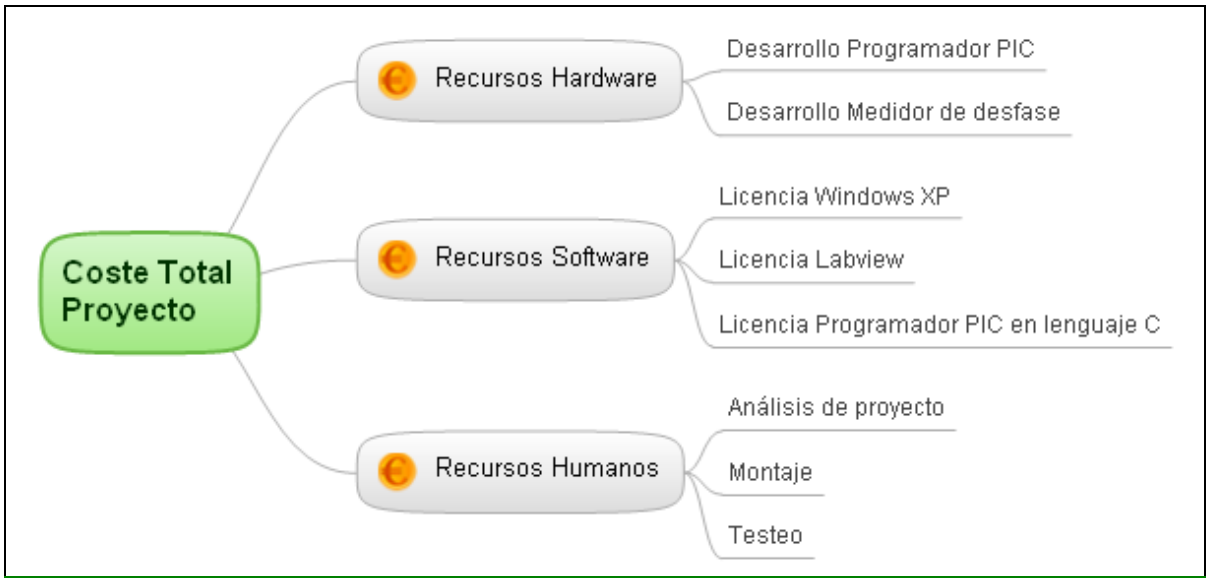


Fig. 23: Diagrama de Costes del proyecto

Recursos Hardware

Desarrollo Programador PIC: ART-2003

Esta parte contiene los componentes empleados en el desarrollo del programador paralelo.

Concepto	Cantidad	Precio por unidad	Precio total
Resistencias	1	0,05 €	0,05 €
Condensadores	1	0,13 €	0,13 €
Zócalo 28 pines para IC	1	0,60 €	0,60 €
Diodos 1N4148	8	0,04 €	
Conector Paralelo D25 Macho	1	2,13 €	2,13 €
Cableado	1 m	0,30 €/m	0,30 €
Placa Puntos paso 2,54 (156x90 mm)	1	4,76 €	4,76 €
		TOTAL:	**7,97 €**

Desarrollo Medidor de desfase

Esta parte contiene tanto los ADCs como el microcontrolador.

Concepto	Cantidad	Precio por unidad	Precio
Resistencias	5	0,05 €	0,25 €
Condensadores cerámicos	5	0,15 €	0,75 €
Condensadores. Electolíticos	2	0,13 €	0,26 €
Pines torneados	30	0,05 €	1,5 €
Cristal 20 MHz	1	0,55 €	0, 55 €
Conector USB tipo B	1	0,73 €	0,73 €
ADC 0801	2	8,74 €	17,48 €
PIC18F2550	1	6 €	6 €
Cableado	5m	0,30 €/m	1,5 €
Placa Puntos paso 2,54 (156x90 mm)	1	4,76 €	4,76 €
		TOTAL:	**29,02 €**

Recursos Software

En esta parte se reflejan las licencias de los programas utilizados para el desarrollo del sistema de adquisición de datos y de la interfaz de cálculo de desfase.

Concepto	Cantidad	Precio por unidad	Precio
Licencia Windows XP Professional	1	284€	284€
Licencia Labview Base	1	1249 €	1249 €
Licencia Compilador CCS PCWH	1	500 €	500 €
		TOTAL:	**2.033 €**

Recursos Humanos

Los recursos humanos tienen en cuenta al aplicar un sueldo de un Ingenierio Junior de 1.500 €/mes.

El concepto de montaje también incluye las pruebas iniciales antes de conseguir el montaje correcto.

Concepto	Horas	Precio por hora	Precio
Análisis	120	9 €	1080 €
Montaje	60	9 €	540 €
Testeo	30	9 €	180 €
		TOTAL:	**1.800 €**

Coste total

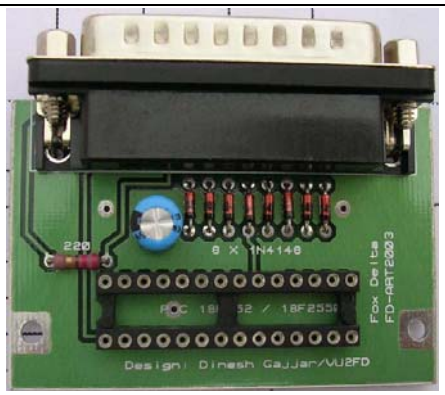
Si sumamos todos los costes, el total queda:

Recursos Hardware	36,99 €
Recursos Software	2.033 €
Recursos Humanos	1.800 €
Total:	3.869,99 €

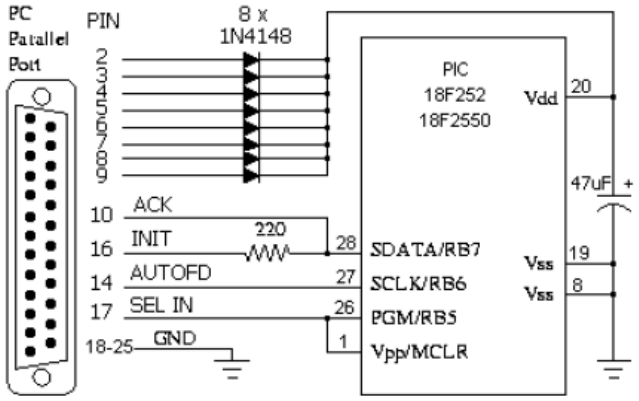
Aunque el coste inicial es alto, la fabricación del equipo en serie puede llegar a bajar el coste total del equipo, pues las licencias de software se abaratan en fabricaciones altas y licencias especificas.

Apéndice B. Programador de microcontrolador PIC: ART-2003

El programador desarrollado es como el que aparece en la figura y que permite programar una gran variedad de microcontroladores de la familia PIC18, y entre ellos el seleccionado PIC18F2550.



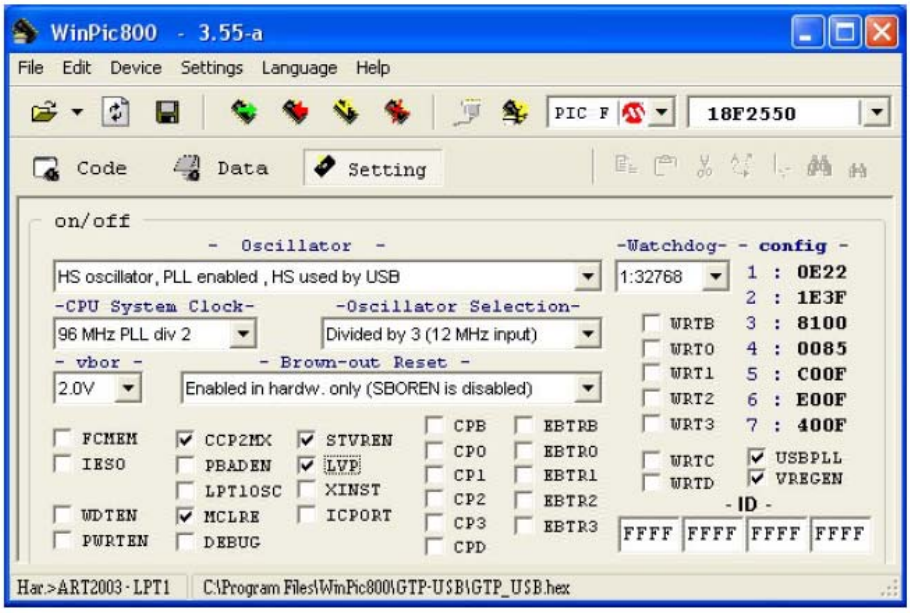
Part	Number	Purpose
U1	28pin IC Socket	Programming Socket
R1	220 ohms	Resistor 1/4W
C1	47-100uf 15V	Electro Capacitor
D1-8	1N4148	Diodes
J1	D25 Male Parallel Connector	R/A PCB Mounted



Foto, Componentes y esquemático del programador ART-2003

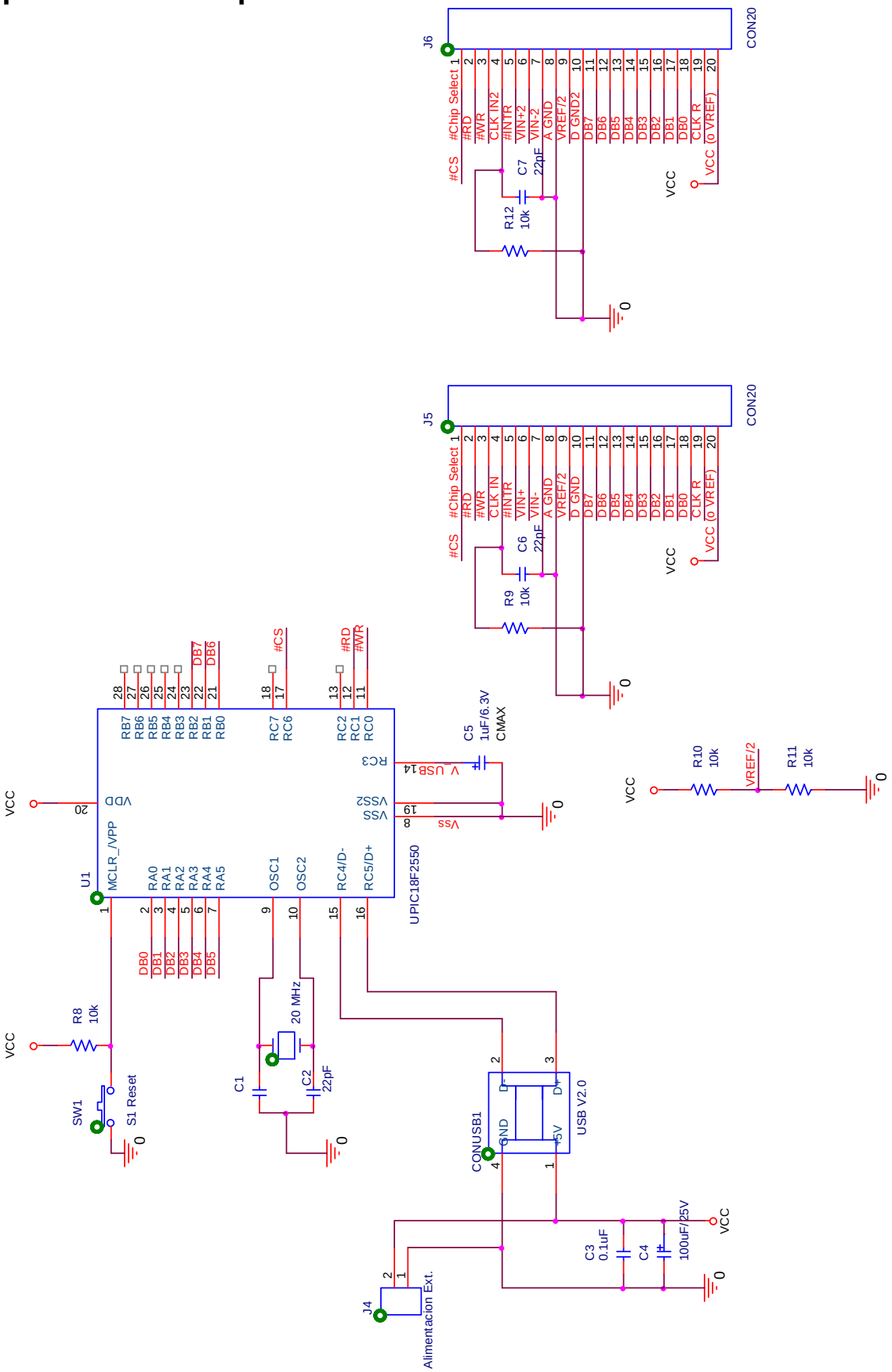
El programador no requiere fuente de alimentación externa y se alimenta directamente con la tensión del puerto paralelo.

Para programar el firmware al microcontrolador, se dispone de un software gratuito llamado WinPic800, como muestra a continuación.



Toda la información de este desarrollo se encuentra gratuitamente en el web:
http://www.foxdelta.com/products/art2003.htm

Apéndice C. Esquema de conexión del microcontrolador con los dispositivos de adquisición de datos.



Apéndice D. Código fuente del programa de adquisición de datos del PIC18F2550

```
////////////////////////////////////////////////////////////////////////
////                                                                ////
////                  ex_usb_serial.c   -Optimizado con ADC         ////
//set to 1 to use a PIC's internal USB Peripheral
//set to 0 to use a National USBN960x peripheral
#define __USB_PIC_PERIF__ 1

#if !defined(__PCH__)
 #error USB CDC Library requires PIC18
#endif

#if __USB_PIC_PERIF__
//  #include <18F2455.h>
  #include <18F2550.h>

  //configure a 20MHz crystal to operate at 48MHz
 #fuses
HSPLL,NOWDT,NOPROTECT,NOLVP,NODEBUG,USBDIV,PLL5,CPUDIV1,VREGEN
 #use delay(clock=48000000)
#else //use the National USBN960x peripheral
  #include <18F452.h>
  #fuses HS,NOWDT,NOPROTECT,NOLVP
  #use delay(clock=20000000)
#endif   //endif check to see which peripheral to use


////////////////////////////////////////////////////////////////////////////
//
// If you are using a USB connection sense pin, define it here.  If you are
// not using connection sense, comment out this line.  Without connection
// sense you will not know if the device gets disconnected.
//       (connection sense should look like this:
//                          100k
//           VBUS-----+----/\/\/\/\----- (I/O PIN ON PIC)
```

```
//                |
//                +----/\/\/\/\-----GND
//                       100k
//       (where VBUS is pin1 of the USB connector)
//
//////////////////////////////////////////////////////////////////////////
///only the 18F4550 development kit has this pin
//#if __USB_PIC_PERIF__ && defined(__PCH__)
// #define USB_CON_SENSE_PIN PIN_B2
//#endif

// Includes all USB code and interrupts, as well as the CDC API
// Includes all USB code and interrupts, as well as the CDC API

/*
PINEADO:
*/

#define AD_WR     PIN_C0    //COMPARTEN AD1 Y AD2
#define AD_RD     PIN_C1    //COMPARTEN AD1 Y AD2
#define AD1_CS    PIN_C6    // PIN 17
#define AD2_CS    PIN_C7    // PIN 18

#define AD_DB0    PIN_A0    //COMPARTEN AD1 Y AD2
#define AD_DB1    PIN_A1    //COMPARTEN AD1 Y AD2
#define AD_DB2    PIN_A2    //COMPARTEN AD1 Y AD2
#define AD_DB3    PIN_A3    //COMPARTEN AD1 Y AD2
#define AD_DB4    PIN_A4    //COMPARTEN AD1 Y AD2
#define AD_DB5    PIN_A5    //COMPARTEN AD1 Y AD2

#define AD_DB6    PIN_B0    //COMPARTEN AD1 Y AD2
#define AD_DB7    PIN_B1    //COMPARTEN AD1 Y AD2

#define PIN_TIME  PIN_B5    // Tiempos envio de muestras
#define PIN_TIMEM PIN_B6    // Tiempo en captura de dos muestras
```

```
#include <usb_cdc.h>
#include <stdlib.h>

void IniciarLectura(int *puertoi);

unsigned int puertox1;
unsigned int puertox2;



void main()
{
char RcvUSB ;
char szDatos[4]={'0', '0', '0', '0'};

int puertoa, puertob;       // Bits del puerto
unsigned long i=0;

long iRetraso = 800;          // Valor por defecto: 800 us
long iMuestras = 300;

//Inicializamos la configuración USB
usb_cdc_init();
usb_init();

// Entramos en un bucle cuando de dispositivo conectado
while(!usb_cdc_connected()) {}
do {
  // Comenzamos a utilizar USB
  usb_task();

  // Comienza nuestro bucle.
  if (usb_enumerated())
  {
    // Capturamos la tecla enviada
    RcvUSB = usb_cdc_getc();
```

```
// Si se trata de una 'B', estamos utilizando el ADC externo
if (RcvUSB=='B')
{
  for (i=0; i<iMuestras; i++)
  {
    output_low(PIN_TIMEM);  // PIN TEST

    // Desctivamos ADC2 y Activamos ADC1
    output_high(AD2_CS);
    output_low(AD1_CS);

    output_high(AD_WR);
    output_high(AD_RD);

    output_low(AD_WR);   // Iniciar Lectura y esperar
    delay_us(10);
    output_high(AD_WR);  // Iniciar Lectura y esperar
    //delay_us(120);
    delay_us(180);
    IniciarLectura(&puertoa);

    // Desactivamos ADC1 y Activamos ADC2
    output_high(AD1_CS);
    output_low(AD2_CS);

    output_high(AD_WR);
    output_high(AD_RD);

    output_low(AD_WR);  // Inicial Lectura y esperar
    delay_us(10);
    output_high(AD_WR); // Inicial Lectura y esperar
    //delay_us(120);
    delay_us(180);
    IniciarLectura(&puertob);

    // Desctivamos ADC1 y ADC2
```

```
      output_high(AD1_CS);
      output_high(AD2_CS);

      // Guardamos los datos a enviar y reseteamos las variables
      puertox1=puertoa;
      puertox2=puertob;
      puertoa=puertob=0;

      // Enviamos los datos
      printf(usb_cdc_putc, "%03u", puertox1);
      printf(usb_cdc_putc, "%03u", puertox2);



      output_high(PIN_TIMEM); // Pin de Test

      // Esperamos el tiempo establecido
      // Sumamos iRetras+tiempo de conversion (PIN_TIMEM)
      delay_us(iRetraso);
   }
   // Fin for que guarda datos en el array

   continue;
} // FIN if ==B

// Configuracion de mustras y tiempo de muestreo
//#####################################################
if (RcvUSB=='N')
{
   RcvUSB = usb_cdc_getc();
   if (RcvUSB=='C')
   {
      szDatos[0] = usb_cdc_getc();
      szDatos[1] = usb_cdc_getc();
      szDatos[2] = usb_cdc_getc();
      szDatos[3] = usb_cdc_getc();
      iMuestras = atol(szDatos);
```

```
            }

            printf(usb_cdc_putc, "\nConfigurado con: \n  - Numero de muestras %Lu.", iMuestras);
            continue;
          }

          if (RcvUSB=='T')
          {
            // el tipo de dato int, por defecto es int8 y por tanto el rango es: -128 a +127
            // Trama F900N250
            RcvUSB = usb_cdc_getc();
            if (RcvUSB=='C')
            {
              szDatos[0] = usb_cdc_getc();
              szDatos[1] = usb_cdc_getc();
              szDatos[2] = usb_cdc_getc();
              szDatos[3] = usb_cdc_getc();

              iRetraso = atol(szDatos);
            }

            printf(usb_cdc_putc, "\nConfigurado con: \n  - Tiempo de muestra %Lu usegundos\n ",
iRetraso);
            continue;
          }

          //continue;
          }

        }while (TRUE);

    }
```

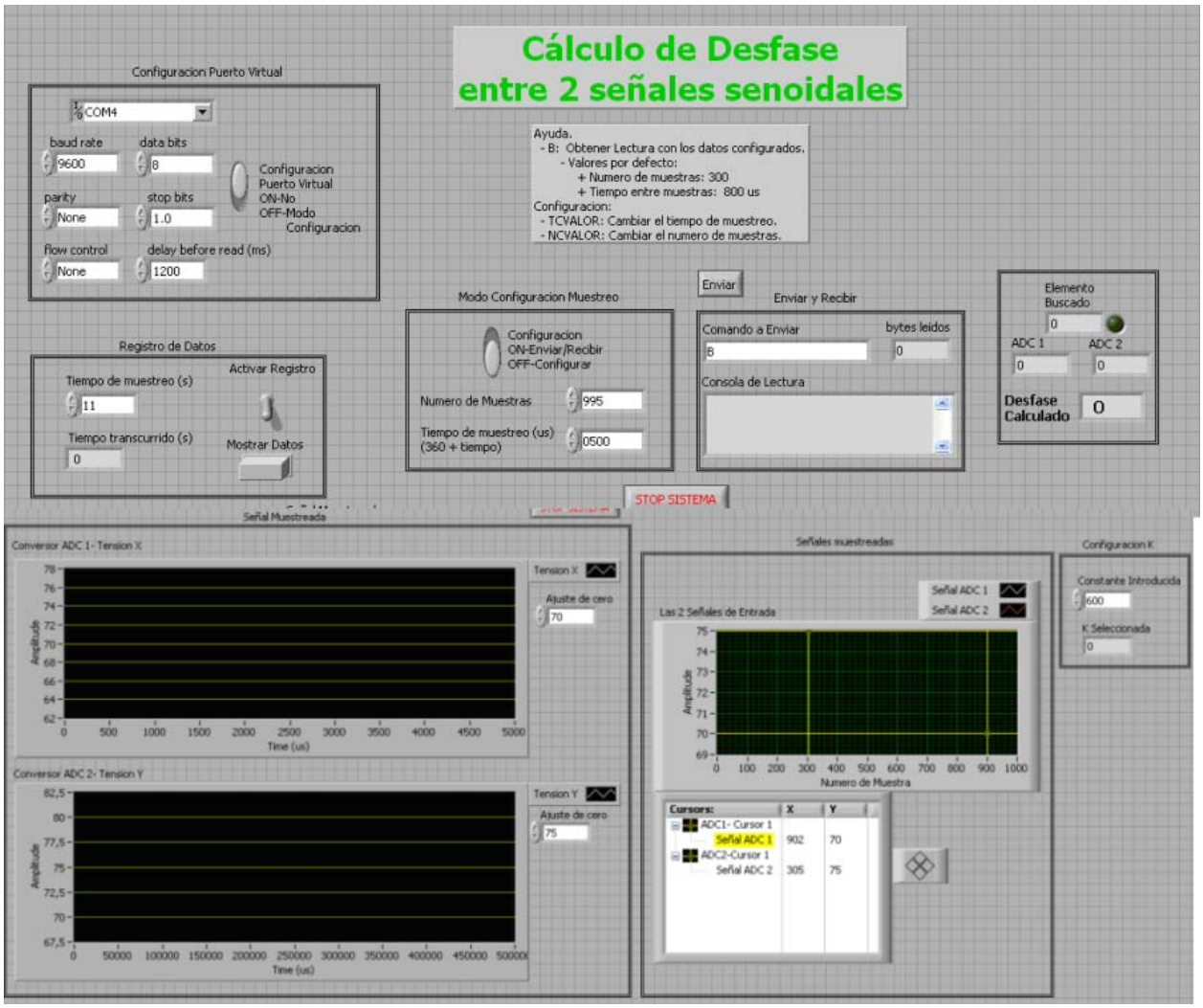
```
void IniciarLectura(int *puertoi)
{
 // Iniciamos la lectura
    output_low(AD_RD);

    // Modificamos
    if(input(AD_DB0)==0)
      bit_clear(*puertoi,0);
    else
      bit_set(*puertoi,0);

    if(input(AD_DB1)==0)
      bit_clear(*puertoi,1);
    else
      bit_set(*puertoi,1);

    if(input(AD_DB2)==0)
      bit_clear(*puertoi,2);
    else
      bit_set(*puertoi,2);

    if(input(AD_DB3)==0)
      bit_clear(*puertoi,3);
    else
      bit_set(*puertoi,3);

    if(input(AD_DB4)==0)
      bit_clear(*puertoi,4);
    else
      bit_set(*puertoi,4);

    if(input(AD_DB5)==0)
      bit_clear(*puertoi,5);
    else
      bit_set(*puertoi,5);
```

```
    if(input(AD_DB6)==0)
      bit_clear(*puertoi,6);
    else
      bit_set(*puertoi,6);

    if(input(AD_DB7)==0)
      bit_clear(*puertoi,7);
    else
      bit_set(*puertoi,7);

}
```

Apéndice E. Código fuente de la interfaz de medición de desfase en Labview

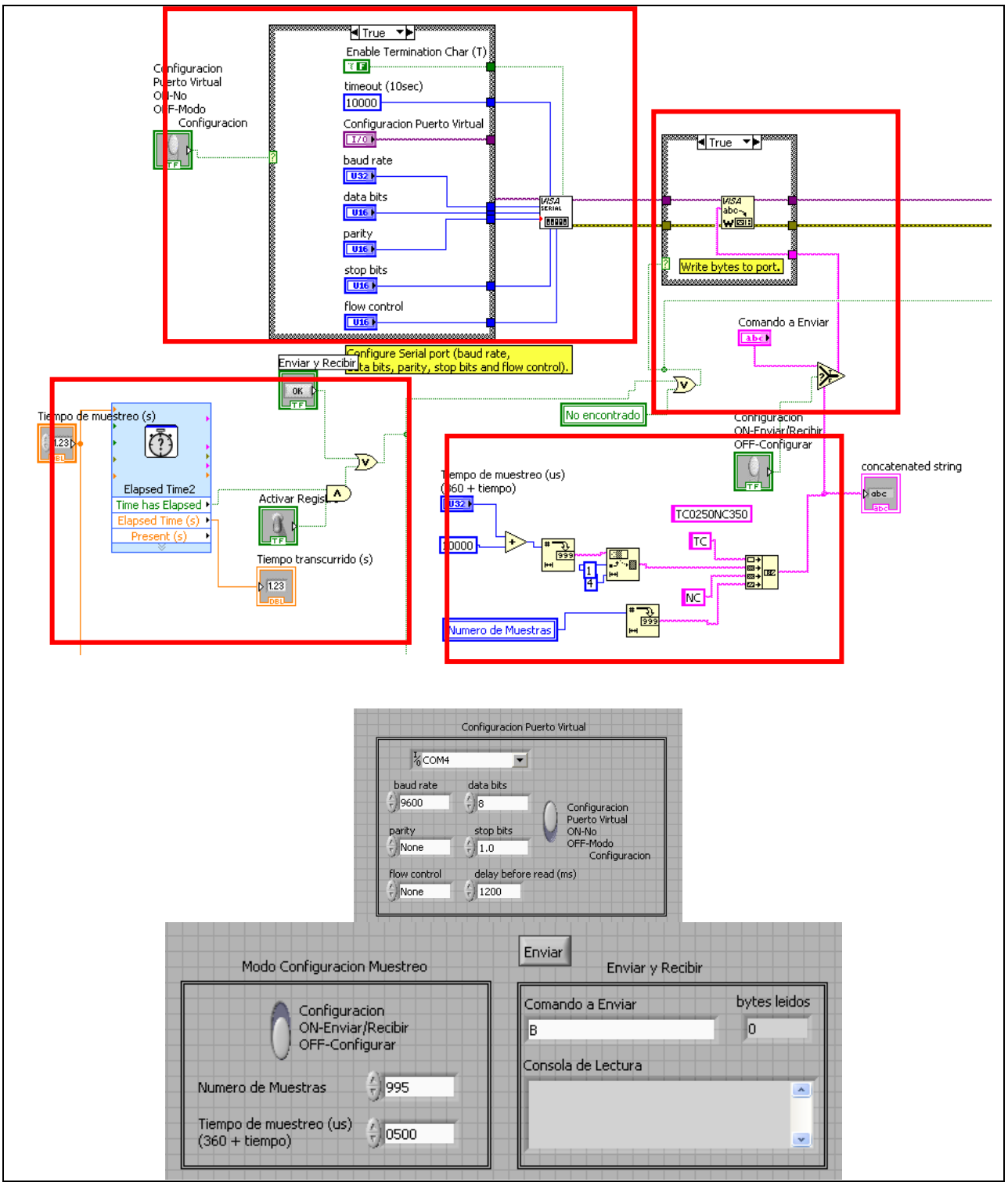
A continuación se muestra una captura de la interfaz de cálculo de desfase. Los bloques se dividen en:

- **Configuración del puerto**: aquí definimos los parámetros básicos del puerto, como velocidad, numero de datos, etc.
- **Registro de datos**: aquí podemos activar el registro de los datos, así como configurar el intervalo de registro.
- **Configuración de muestreo**: aquí podemos definir la información de número de muestras y frecuencia de muestreo que queremos utilizar.
- **Envío de comandos y consola de recepción de datos**: desde aquí nos comunicamos con el microcontrolador
- **Gráfico de señales de entrada separadas y gráfico de señales conjuntas.**
- **Configuración de K**: aquí es donde se introduce el valor que representa media onda.
- **Resultado del calculo de desfase**: panel que visualiza el desfase calculado.



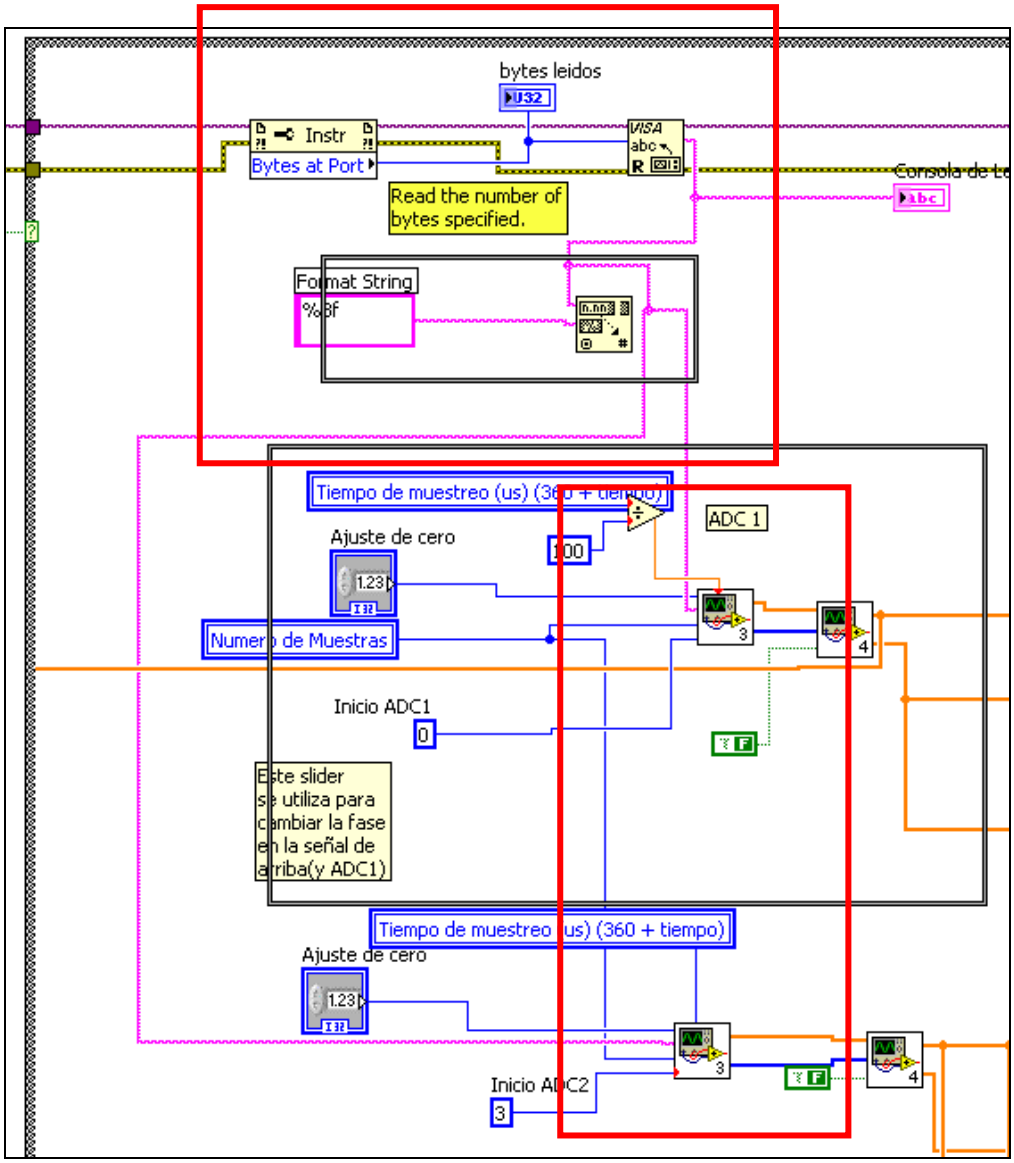
Bloque de configuración de muestreo, envío de comandos/recepción de datos.

En el primer bloque se configura el puerto virtual como aparece en la parte superior de la imagen. En la parte derecha de la imagen se encuentra la configuración del muestreo de datos. En la parte superior derecha, se encuentra en bloque de envío de datos. En la parte inferior izquierda se encuentra una de las condiciones que habilita el envío de comandos siempre y cuando esté habilitado el registro de datos.



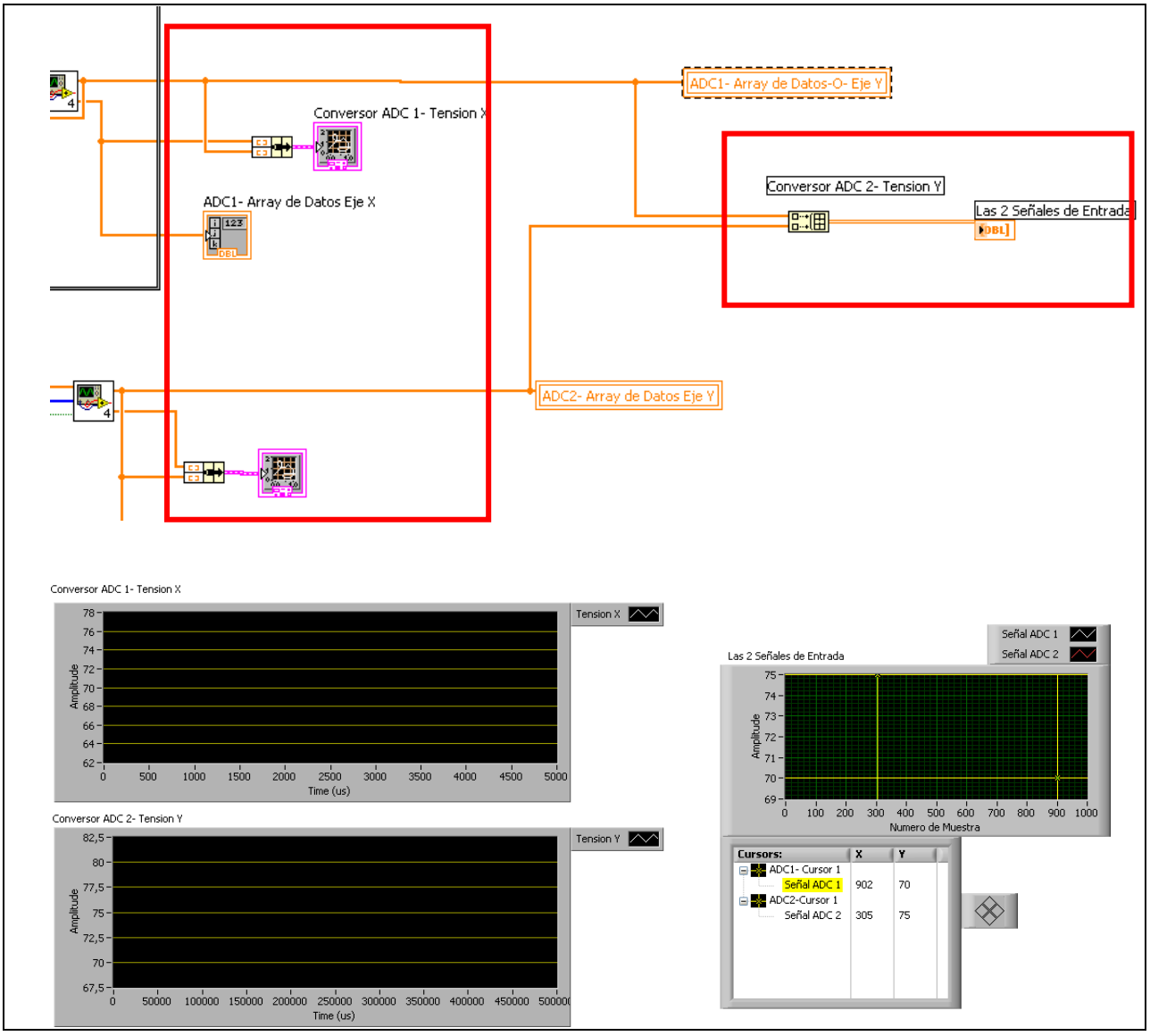
Bloque de adquisición de datos y almacenaje de los datos de las señales en arrays.

En los bloque de la parte superior de la imagen se encuentra la adquisición de datos del puerto. En ese momento se van guardando los datos en dos arrays para cada una de las señales analógicas.



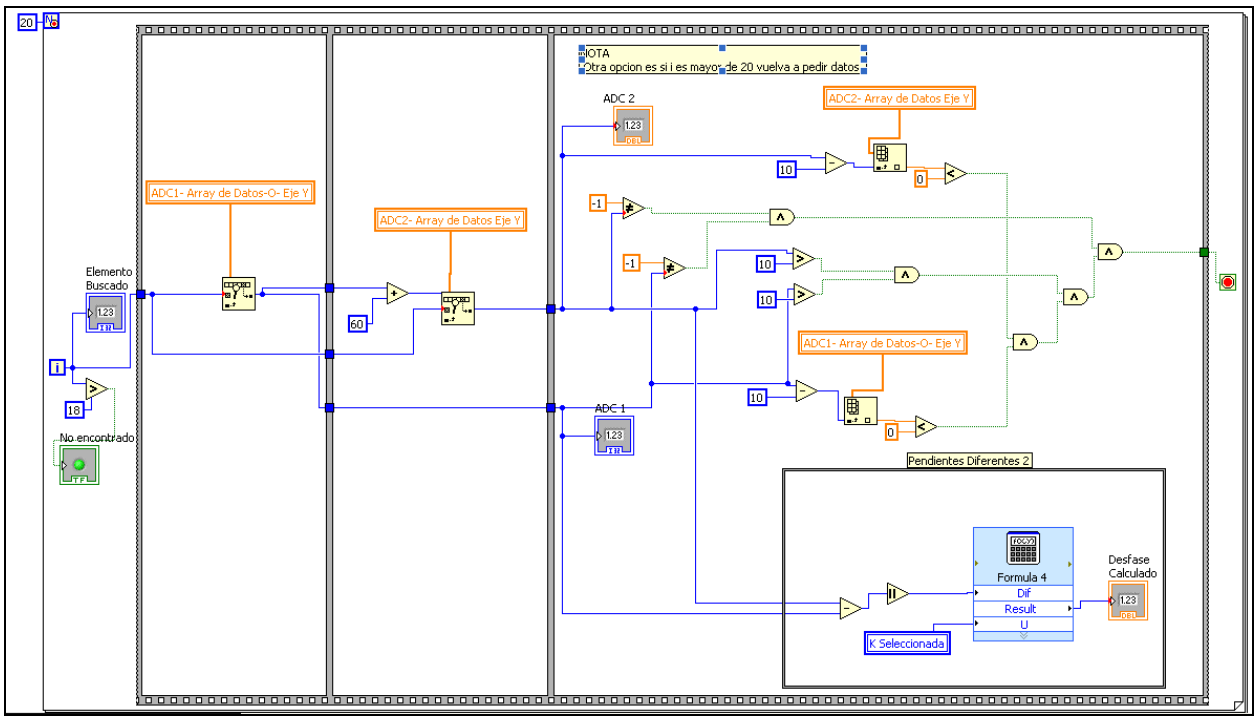
Bloque de visualización de graficas

Estos bloques cogen los datos de los arrays de datos de los ADCs y se visualizan en dos gráficas por separado y en una de forma conjunta como aparece a la derecha de la imagen.



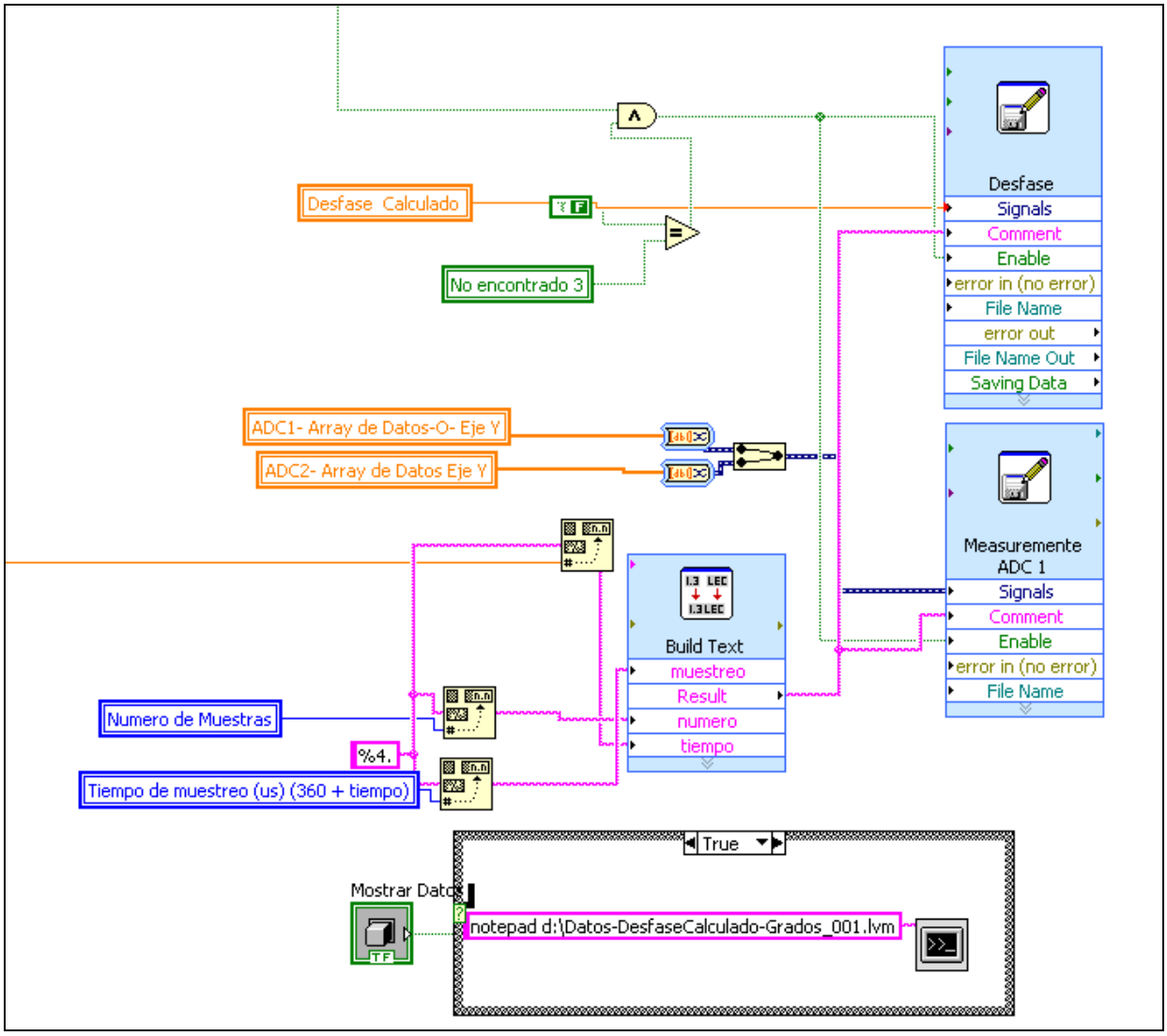
Bloque que cálculo del desfase de las señales

Este bloque calcula el desfase de las señales y lo visualiza en pantalla.



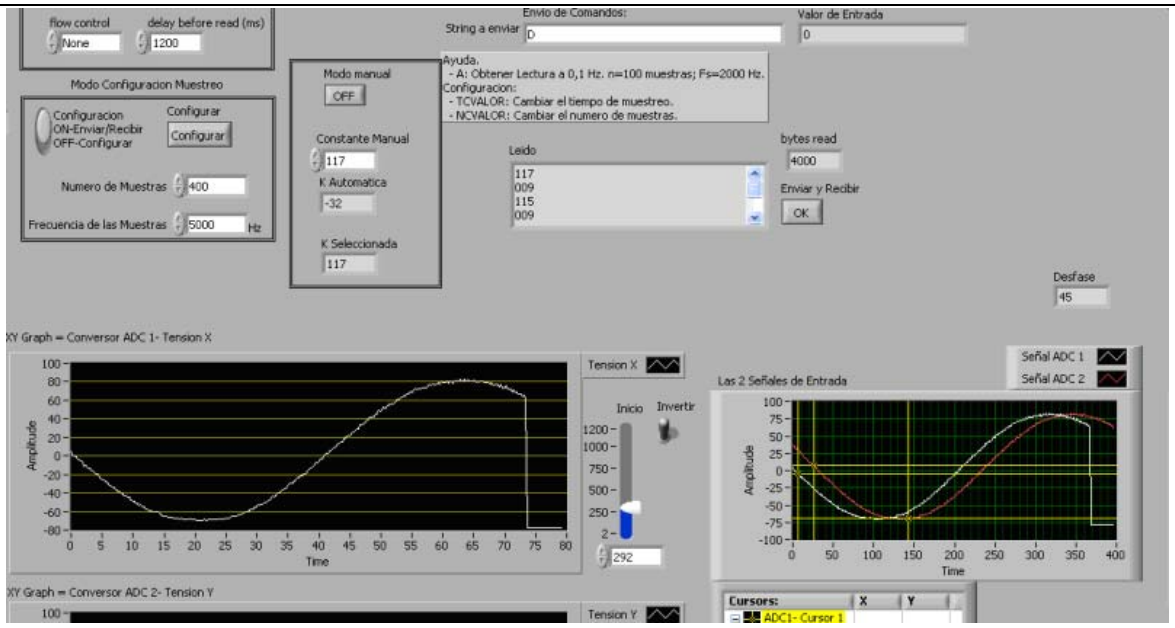
Bloque de generación de log

Este bloque construye el archivo con los datos del desfase y por último lo envía a disco, siempre y cuando se encuentre configurado el registro.

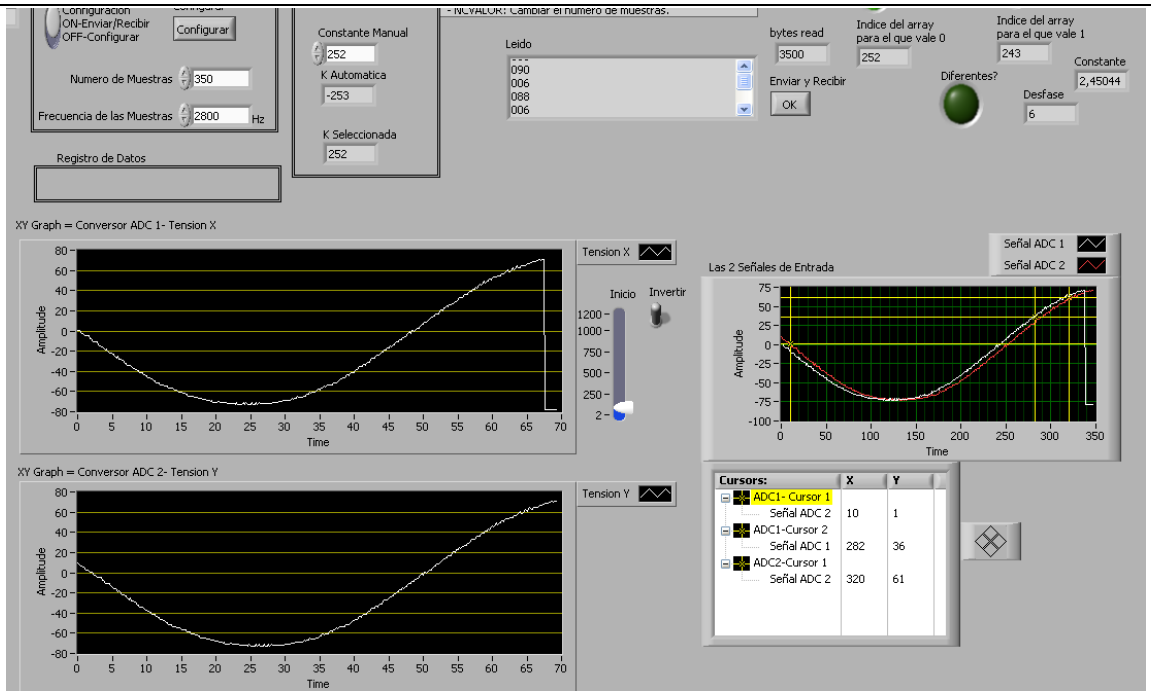


Apéndice F. Pruebas de medición de desfase realizadas con la interfaz.

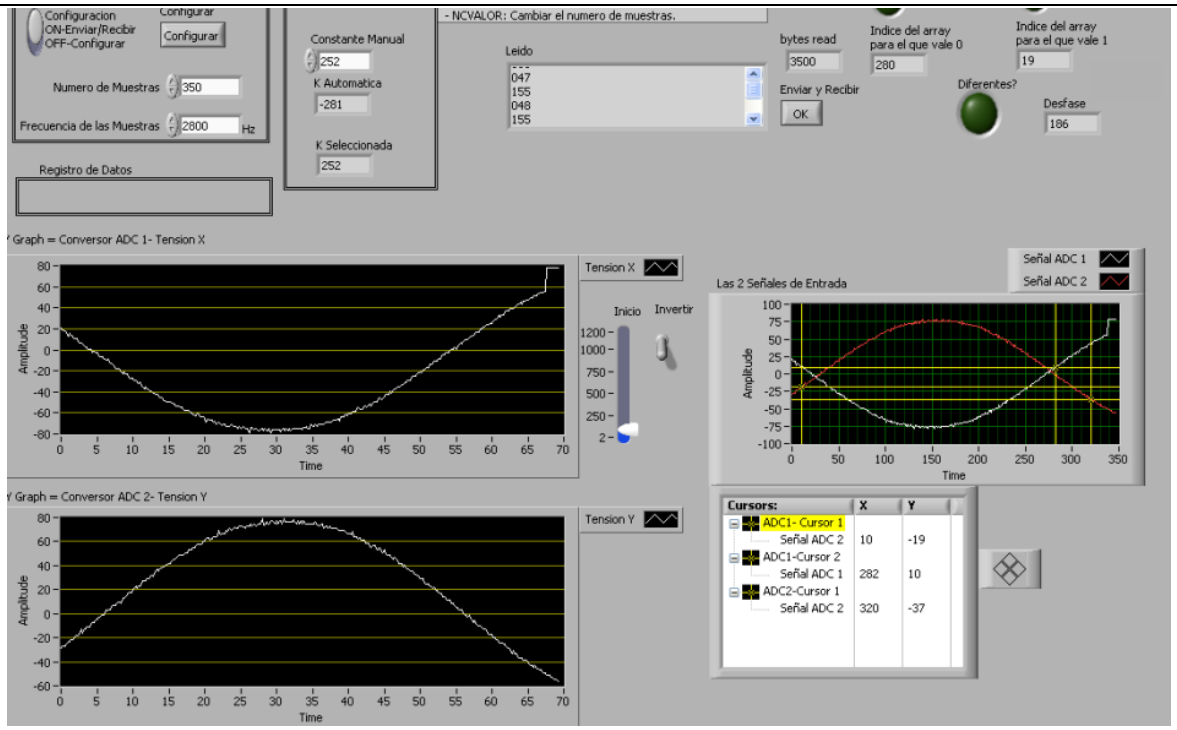
En este apéndice se muestran las diferentes pruebas que se han realizado de cálculo del desfase a diferentes frecuencias. Las capturas se realizaron en diferentes fases del desarrollo de la interfaz, por lo que las imágenes contienen diferentes bloques.



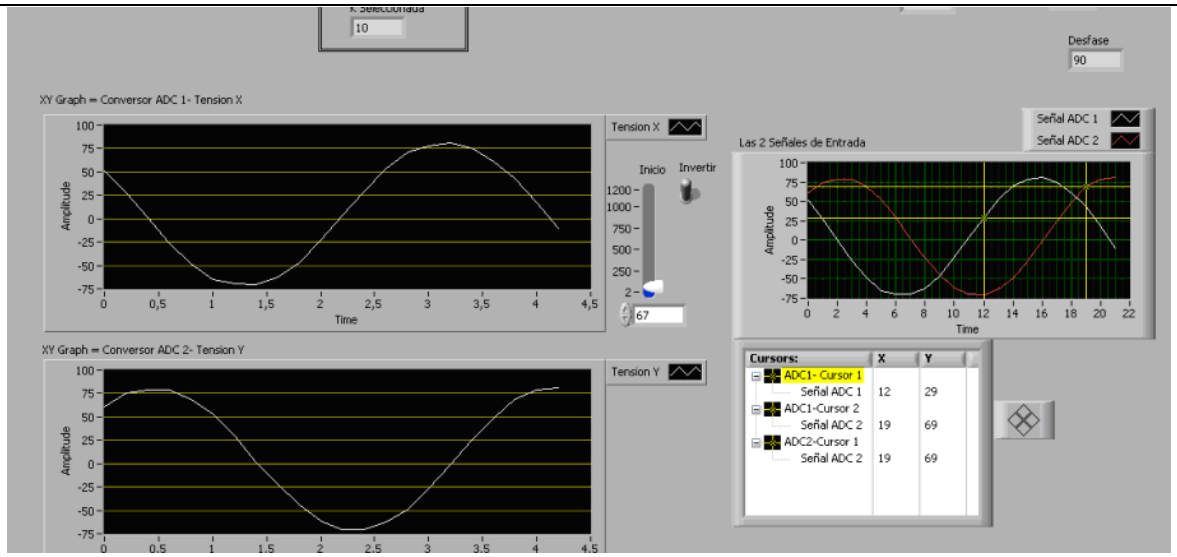
Frecuencia 1 Hz, desfase 45 grados



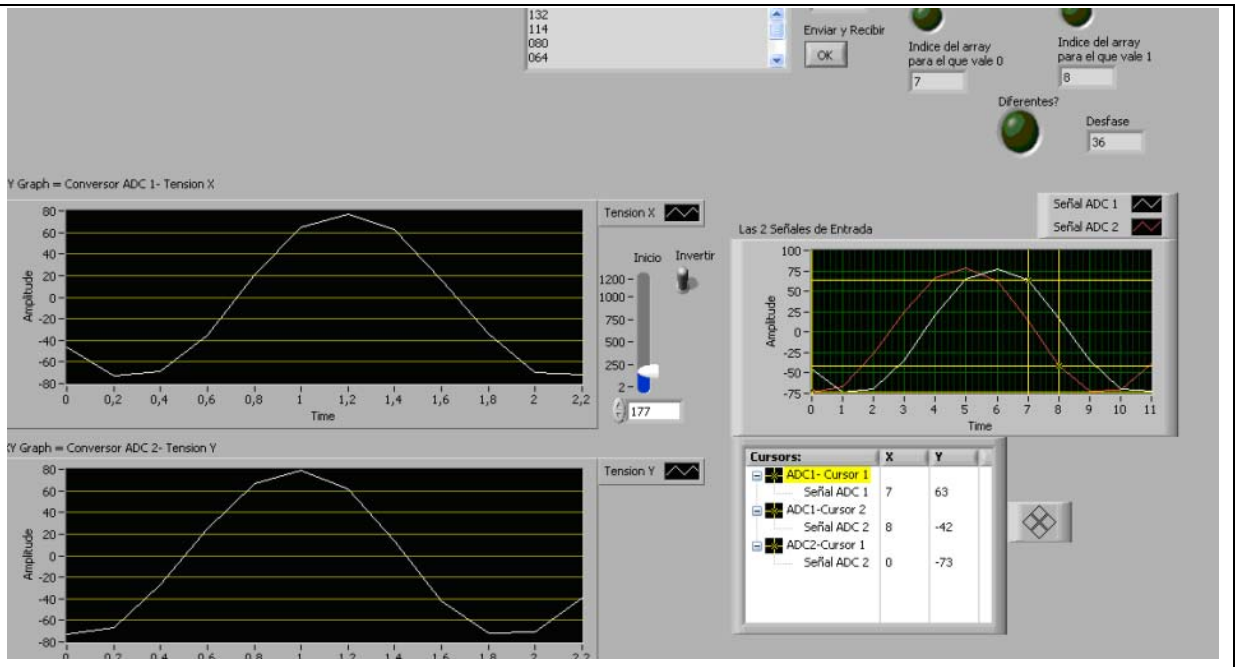
Frecuencia 1,6 Hz, desfase 6 grados



Frecuencia 1,6 Hz, desfase 186 grados



Frecuencia 127 Hz, desfase 90 grados

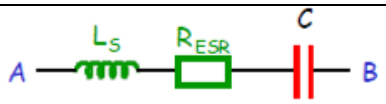


Frecuencia 500 Hz, desfase 36 grados

Apéndice G. Variación de la tg δ en un condensador

Este apéndice tiene como objetivo describir brevemente el significado de la tg δ que la que se desea medir como aplicación práctica de la interfaz de medida de desfase de dos señales senoidales.

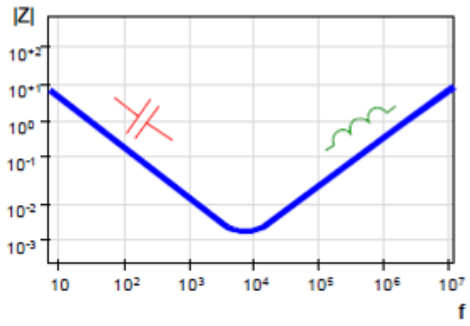
Al cortocircuitar los bornes del transformador, éste se comporta como un condensador, cuyo circuito equivalente se muestra en la figura.



$$Z = R_{ESR} + j\cdot\omega\cdot L_S - j\cdot\frac{1}{\omega\cdot C}$$

Circuito equivalente de un condensador

La frecuencia utilizada para la medida debe ser baja, pues como se puede observar en el circuito, el valor de la bobina se desprecia frente al valor del condensador que aumenta.



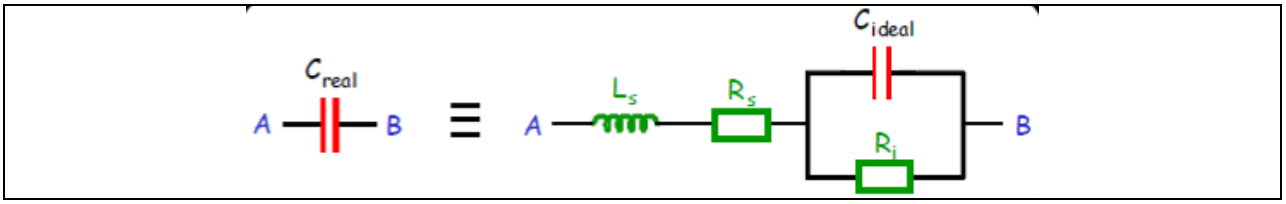
- **A bajas frecuencias**: comportamiento es esencialmente capacitivo.
- **A altas frecuencias**: el comportamiento inductivo es el que predomina.
- **A frecuencias próximas a la frecuencia de resonancia**: la impedancia es R_{ESR}.

Pero si profundizamos más en el circuito equivalente del condensador, nos permite definir el coeficiente de pérdidas, cuya fórmula se muestra a continuación:

$$tg\delta = \frac{P}{Q} = \frac{1}{2\cdot\pi\cdot f\cdot C\cdot R_i} + 2\cdot\pi\cdot f\cdot C\cdot R_s$$

Trabajando a baja frecuencia, el efecto predominante es el de la resistencia R_i, mientras que a frecuencias elevadas es R_s la que predomina.

R_i hace precisamente que varíe el valor de la tg δ y que produce una variación del dieléctrico del condensador.



Como curiosidad, decir que el valor de **L_S y R_S** se debe a los contactos del dieléctrico, mientras que **R_i** como hemos hablado se debe al dieléctrico.fdfdf

Apéndice H. Hojas de catálogo

National Semiconductor

August 2000

LM124/LM224/LM324/LM2902
Low Power Quad Operational Amplifiers

General Description

The LM124 series consists of four independent, high gain, internally frequency compensated operational amplifiers which were designed specifically to operate from a single power supply over a wide range of voltages. Operation from split power supplies is also possible and the low power supply current drain is independent of the magnitude of the power supply voltage.

Application areas include transducer amplifiers, DC gain blocks and all the conventional op amp circuits which now can be more easily implemented in single power supply systems. For example, the LM124 series can be directly operated off of the standard +5V power supply voltage which is used in digital systems and will easily provide the required interface electronics without requiring the additional ±15V power supplies.

Unique Characteristics

- In the linear mode the input common-mode voltage range includes ground and the output voltage can also swing to ground, even though operated from only a single power supply voltage
- The unity gain cross frequency is temperature compensated
- The input bias current is also temperature compensated

Advantages

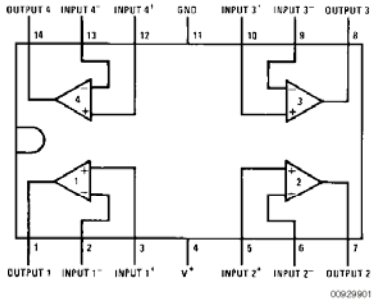
- Eliminates need for dual supplies
- Four internally compensated op amps in a single package
- Allows directly sensing near GND and V_{OUT} also goes to GND
- Compatible with all forms of logic
- Power drain suitable for battery operation

Features

- Internally frequency compensated for unity gain
- Large DC voltage gain 100 dB
- Wide bandwidth (unity gain) 1 MHz (temperature compensated)
- Wide power supply range:
  Single supply 3V to 32V
  or dual supplies ±1.5V to ±16V
- Very low supply current drain (700 µA)—essentially independent of supply voltage
- Low input biasing current 45 nA (temperature compensated)
- Low input offset voltage 2 mV and offset current: 5 nA
- Input common-mode voltage range includes ground
- Differential input voltage range equal to the power supply voltage
- Large output voltage swing 0V to V+ − 1.5V

Connection Diagrams

Dual-In-Line Package



Top View

Order Number LM124J, LM124AJ, LM124J/883 (Note 2), **LM124AJ/883** (Note 1), **LM224J, LM224AJ, LM324J, LM324M, LM324MX, LM324AM, LM324AMX, LM2902M, LM2902MX, LM324N, LM324AN, LM324MT, LM324MTX or LM2902N LM124AJRQML and LM124AJRQMLV**(Note 3)
See NS Package Number J14A, M14A or N14A

MICROCHIP

PIC18F2455/2550/4455/4550

28/40/44-Pin, High-Performance, Enhanced Flash, USB Microcontrollers with nanoWatt Technology

Universal Serial Bus Features:

- USB V2.0 Compliant
- Low Speed (1.5 Mb/s) and Full Speed (12 Mb/s)
- Supports Control, Interrupt, Isochronous and Bulk Transfers
- Supports up to 32 Endpoints (16 bidirectional)
- 1-Kbyte Dual Access RAM for USB
- On-Chip USB Transceiver with On-Chip Voltage Regulator
- Interface for Off-Chip USB Transceiver
- Streaming Parallel Port (SPP) for USB streaming transfers (40/44-pin devices only)

Power-Managed Modes:

- Run: CPU on, peripherals on
- Idle: CPU off, peripherals on
- Sleep: CPU off, peripherals off
- Idle mode currents down to 5.8 µA typical
- Sleep mode currents down to 0.1 µA typical
- Timer1 Oscillator: 1.1 µA typical, 32 kHz, 2V
- Watchdog Timer: 2.1 µA typical
- Two-Speed Oscillator Start-up

Flexible Oscillator Structure:

- Four Crystal modes, including High Precision PLL for USB
- Two External Clock modes, up to 48 MHz
- Internal Oscillator Block:
  - 8 user-selectable frequencies, from 31 kHz to 8 MHz
  - User-tunable to compensate for frequency drift
- Secondary Oscillator using Timer1 @ 32 kHz
- Dual Oscillator options allow microcontroller and USB module to run at different clock speeds
- Fail-Safe Clock Monitor:
  Allows for safe shutdown if any clock stops

Peripheral Highlights:

- High-Current Sink/Source: 25 mA/25 mA
- Three External Interrupts
- Four Timer modules (Timer0 to Timer3)
- Up to 2 Capture/Compare/PWM (CCP) modules:
  - Capture is 16-bit, max. resolution 5.2 ns (TCY/16)
  - Compare is 16-bit, max. resolution 83.3 ns (TCY)
  - PWM output: PWM resolution is 1 to 10-bit
- Enhanced Capture/Compare/PWM (ECCP) module:
  - Multiple output modes
  - Selectable polarity
  - Programmable dead time
  - Auto-shutdown and auto-restart
- Enhanced USART module:
  - LIN bus support
- Master Synchronous Serial Port (MSSP) module supporting 3-wire SPI (all 4 modes) and I²C™ Master and Slave modes
- 10-bit, up to 13-channel Analog-to-Digital Converter module (A/D) with Programmable Acquisition Time
- Dual Analog Comparators with Input Multiplexing

Special Microcontroller Features:

- C Compiler Optimized Architecture with optional Extended Instruction Set
- 100,000 Erase/Write Cycle Enhanced Flash Program Memory typical
- 1,000,000 Erase/Write Cycle Data EEPROM Memory typical
- Flash/Data EEPROM Retention: > 40 years
- Self-Programmable under Software Control
- Priority Levels for Interrupts
- 8 x 8 Single-Cycle Hardware Multiplier
- Extended Watchdog Timer (WDT):
  Programmable period from 41 ms to 131s
- Programmable Code Protection
- Single-Supply 5V In-Circuit Serial Programming™ (ICSP™) via two pins
- In-Circuit Debug (ICD) via two pins
- Optional dedicated ICD/ICSP port (44-pin devices only)
- Wide Operating Voltage Range (2.0V to 5.5V)

Device	Program Memory		Data Memory		I/O	10-Bit A/D (ch)	CCP/ECCP (PWM)	SPP	MSSP		EAUSART	Comparators	Timers 8/16-Bit
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)					SPI	Master I²C™			
PIC18F2455	24K	12288	2048	256	24	10	2/0	No	Y	Y	1	2	1/3
PIC18F2550	32K	16384	2048	256	24	10	2/0	No	Y	Y	1	2	1/3
PIC18F4455	24K	12288	2048	256	35	13	1/1	Yes	Y	Y	1	2	1/3
PIC18F4550	32K	16384	2048	256	35	13	1/1	Yes	Y	Y	1	2	1/3

Preliminary

National Semiconductor

November 1999

ADC0801/ADC0802/ADC0803/ADC0804/ADC0805
8-Bit µP Compatible A/D Converters

General Description

The ADC0801, ADC0802, ADC0803, ADC0804 and ADC0805 are CMOS 8-bit successive approximation A/D converters that use a differential potentiometric ladder—similar to the 256R products. These converters are designed to allow operation with the NSC800 and INS8080A derivative control bus with TRI-STATE output latches directly driving the data bus. These A/Ds appear like memory locations or I/O ports to the microprocessor and no interfacing logic is needed.

Differential analog voltage inputs allow increasing the common-mode rejection and offsetting the analog zero input voltage value. In addition, the voltage reference input can be adjusted to allow encoding any smaller analog voltage span to the full 8 bits of resolution.

Features

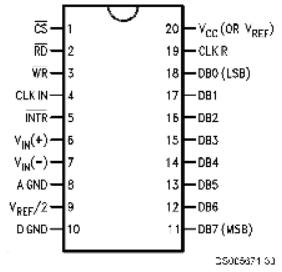
- Compatible with 8080 µP derivatives—no interfacing logic needed - access time - 135 ns
- Easy interface to all microprocessors, or operates "stand alone"
- Differential analog voltage inputs
- Logic inputs and outputs meet both MOS and TTL voltage level specifications
- Works with 2.5V (LM336) voltage reference
- On-chip clock generator
- 0V to 5V analog input voltage range with single 5V supply
- No zero adjust required
- 0.3" standard width 20-pin DIP package
- 20-pin molded chip carrier or small outline package
- Operates ratiometrically or with 5 V_{DC}, 2.5 V_{DC}, or analog span adjusted voltage reference

Key Specifications

- Resolution 8 bits
- Total error ±¼ LSB, ±½ LSB and ±1 LSB
- Conversion time 100 µs

Connection Diagram

ADC080X
Dual-In-Line and Small Outline (SO) Packages



DS005671-1

See Ordering Information

Ordering Information

TEMP RANGE		0°C TO 70°C	0°C TO 70°C	−40°C TO +85°C
ERROR	±¼ Bit Adjusted			ADC0801LCN
	±½ Bit Unadjusted	ADC0802LCWM		ADC0802LCN
	±½ Bit Adjusted			ADC0803LCN
	±1Bit Unadjusted	ADC0804LCWM	ADC0804LCN	ADC0805LCN/ADC0804LCJ
PACKAGE OUTLINE		M20B—Small Outline	N20A—Molded DIP	

Z-80® is a registered trademark of Zilog Corp

Apéndice I. Documentación del CD

El CD contiene la siguiente información

- Datasheets:
  - Medidor del ángulo de desfase PAM360™ (GE Enery)
  - Sistema de diagnóstico del aislamiento IDA200 ™ (GE Enery)
  - Preamplificador de corriente modelo 5182 (Signal Recovery)
  - Microcontrolador PIC18F2550 (Microchip)
  - Convertidor A/D de 8 bits AD0801 (National Semiconductor)
  - Amplificador operacional de baja potencia LM324 (National Semiconductor)
- Manuales:
  - Manual de usuario de LabVIEW (National Semiconductor)
  - Manual de usuario del compilador PCW de CCS®
    - Autor: Andrés Cánovas López
- Esquemas:
  - Circuito PIC18F2550 y ADCs
  - Circuito acondicionador de pruebas.
- Código fuente:
  - Firmware de PIC18F2550
  - Código fuente de la interfaz en Labview